

Name_____

Please circle your section number:

010 (Wed PM)

011 (Mon PM)

012 (Mon AM)

Answer the multiple choice questions on a “Scantron Form”

Bubble in ONLY your Unix userid and your answers

DO NOT bubble in your id number or section

If you bubble in your SSN, the computer will **reject your form!!!**

Answer the remaining questions directly on the exam paper.

General Instructions

- The exam is @@@% multiple choice, and @@@% programming.
- The programming questions start with number 21. You may want to tackle them first, since they may take more time.
- DO NOT WRITE YOUR NAME ON ANY PAGE EXCEPT THIS ONE!
- You have 50 minutes. **Pace yourself**, and pay attention to the point values.
- Read *all* the directions *carefully* on each problem.
- Good luck.

Questions 1 through 4 deal with the following code excerpt:

```
1 // foo.cc
2
3 #include <iostream>
4 using namespace std;
5
6 int main(void)
7 {
8     char mascot[10]="Blue Hens";
9     char hometown[20]="Newark";
10    char school[30]="University of Delaware";
11    char state[20]="Delaware";
12
13    char *a = mascot;
14    char * const c = hometown;
15    const char *b = school;
16    const char * const d = state;
17
18
19    cout << a << endl;
20    cout << b << endl;
21    cout << c << endl;
22    cout << d << endl;
23
24    // where some extra code might go
25
26 #include "extra.h"
27
28    cout << a << endl;
29    cout << b << endl;
30    cout << c << endl;
31    cout << d << endl;
32
33 }
34
35
36
37
38
39
40
41
42
43
44
45
46
```

1. (2 pts) One of these lines of code would produce a compiler error if placed at the spot indicated by // where some extra code might go ?
Which one is WRONG?
 - (a) `a[0] = 'F';`
 - (b) `b = hometown;`
 - (c) `a[1] = 'r';`
 - (d) `b[0] = 'D';`
 - (e) `a = hometown;`

2. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?
Which one is WRONG?
 - (a) `a = mascot;`
 - (b) `a = state;`
 - (c) `b[1] = 'u';`
 - (d) `c[1] = 'm';`
 - (e) `a[1] = 'u';`

3. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?
Which one is WRONG?
 - (a) `d = state;`
 - (b) `b = hometown;`
 - (c) `c[0] = 'D';`
 - (d) `b = mascot`
 - (e) `c[1] = 'i';`

4. (2 pts) Read carefully! Only one of these lines of code would NOT produce a compiler error if placed at the spot indicated by // where some extra code might go ?
That is, which one is CORRECT?
 - (a) `c[1] = 'u';`
 - (b) `a[1] = 'r';`
 - (c) `c = d;`
 - (d) `c = a;`
 - (e) `d = hometown;`

Questions 5 through 8 refer to the fragment of code shown in Figure 1 that comes from a complete C++ program.

Figure 1: Declaration of a struct

```
...
struct Node
{
    int data;
    Node *next;
};

int main(void)
{
    Node *p;
    Node s;
    p = new Node;
    ...
}
```

5. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 1, at or after the line indicated by . . .
Which one?
- (a) `data = 3`
 - (b) `s.data = 3;`
 - (c) `(*s).data = 3;`
 - (d) `s->data = 3;`
 - (e) `p.data = 3;`
6. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 1, at or after the line indicated by . . .
Which one?
- (a) `next = NULL;`
 - (b) `p->next = NULL;`
 - (c) `s->next = NULL;`
 - (d) `p.next = NULL;`
 - (e) `(*s).next = NULL;`

7. (2 pts) Which of the following is equivalent to `p->data`?
- (a) `(*p).data`
 - (b) `*(p.data)`
 - (c) `p.(*data)`
 - (d) `p[data]`
 - (e) `p[1].data`
8. (2 pts) Which of the following is equivalent to `p->next`? (Hint: This is is tricky: think carefully about the equivalence of array subscripts and dereferencing of pointers.)
- (a) `*(p.next)`
 - (b) `p[next]`
 - (c) `p[0].next`
 - (d) `p[1].next`
 - (e) `*p[next]`
9. (2 pts) Suppose you want to write a function `incrModFive` that will increment an integer mod 5 (i.e. if the integer is 4 and you increment it, it will return to 0.) If you want the function `incrModFive` to take one parameter, an `int` passed by reference, and return nothing at all, which of the following would be a correct function prototype for `incrModFive`?
- (a) `incrModFive(int x);`
 - (b) `void incrModFive(int x);`
 - (c) `void incrModFive(int &x);`
 - (d) `int incrModFive(int x);`
 - (e) `incrModFive (int &x);`
10. (2 pts) (Continuation of previous question.) If you then called that function, which of the following is a legal call to function `incrModFive`? Assume that the following declarations are in effect:
- ```
int y,z;
y = 3;
z = 4;
int *x;
x = &y;
```
- (a) `incrModFive(&z);`
  - (b) `incrModFive(y);`
  - (c) `incrModFive(x);`
  - (d) `incrModFive(*y);`
  - (e) `z=incrModFive(4);`

11. (2 pts) If the variable `a` refers to an array of 10 elements, then the expression `a[0]` is equivalent to which of the following?
- (a) `a`
  - (b) `*a`
  - (c) `a+0`
  - (d) `a++`
  - (e) `++a`

12. (2 pts) Suppose two function definitions in the same program have the same name, but differ in the number and/or type of their formal parameters, as illustrated by these two function prototypes:

```
int max(int a, int b);
int max(int a, int b, int c);
```

This would be an example of which of the following?

- (a) a binary scope resolution operator
  - (b) function overloading
  - (c) a unary scope resolution operator
  - (d) separate compilation
  - (e) a syntax error, since this is not permitted
13. (2 pts) If you want the variable `a` to contain the address of an `double`, which of the following would be a correct declaration of `a`?
- (a) `double a;`
  - (b) `double &a;`
  - (c) `double *a;`
  - (d) `double %a;`
  - (e) none of the above
14. (2 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?
- (a) `*p`
  - (b) `&p`
  - (c) `deref(p)`
  - (d) `%p`
  - (e) `$p`

15. (2 pts) Suppose a UD student named Terry Bly Smart has the userid tbsmart. Which of the following URLs brings up his/her personal web page on strauss?

(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)

- (a) `http://strauss.udel.edu/tbsmart`
- (b) `http://strauss.udel.edu/~tbsmart`
- (c) `http://strauss.udel.edu/~tbsmart/public.html`
- (d) `http://strauss.udel.edu/~tbsmart/public.html`

16. (2 pts) Suppose in a future semester, your instructor for CISC321 requires you to create a web page on strauss with the URL

`http://strauss.udel.edu/~userid/CISC321`. Which of the following files could you edit to create the web page?

(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)

- (a) `~/CISC321/index.html`
- (b) `~/public_html/CISC321/index.html`
- (c) `~/public_html/cisc321/index.html`
- (d) `~/public_html/CISC321/index.html`

Here is a sample program, followed by some sample output.—except, the numbers have been replaced with blanks First, trace through the program to determine what goes in each of these blanks, then select the correct answers from the choices given in the problems.

**Don't panic:** Each of the sections of code can be done independently of the other sections, so if you are stuck on one, don't stop; keep working.

```
1 /* ql0.cc Agnes Nitt for CISC181 final exam */
2
3 #include <iostream>
4 using namespace std;
5
6 void trouble(int *r);
7 double fire(double f);
8 int burn(int b, int *n);
9 void cauldron(int &c, int &d);
10
11 int main(void)
12 {
13 int e=2, f=3, t = 4, y = 9;
14 double v=5.1, z = 10.2;
15
16 trouble(&y);
17 cout << "y=" << y << endl;
18
19 z = fire(v);
20 cout << "v=" << v << " z=" << z << endl;
21
22 t = 3;
23 burn(5, &t);
24 cout << "t=" << t << endl;
25
26 e=4;
27 f=5;
28 cauldron(e,f);
29 cout << "e=" << e << " f=" << f << endl;
30 return 0;
31 }
32
33 void trouble(int *r)
34 {
35 (*r) = -1;
36 return;
37 }
38
39 void cauldron(int &c, int &d)
40 {
41 int temp = c;
42 c = d;
43 d = temp;
44 }
45
46 int burn(int b, int *n)
47 {
48 (*n) = b;
49 }
50
51 double fire(double f)
52 {
53 return (f + 0.1);
54 }
```

```
> ./q10
y=___
v=___ z=___
t=___
e=___ f=___
>
```

17. (2 pts) What goes in the blank  $y=$ \_\_\_ ?

- (a)  $y=-1$
- (b)  $y=-9$
- (c)  $y=9$
- (d)  $y=-8$
- (e) none of the above

18. (2 pts) What goes in the blank  $v=$ \_\_\_  $z=$ \_\_\_ ?

- (a)  $v=5.1$   $z=5.1$
- (b)  $v=5.2$   $z=5.2$
- (c)  $v=5.2$   $z=5.1$
- (d)  $v=5.1$   $z=5.2$
- (e) none of the above

19. (2 pts) What goes in the blank  $t=$ \_\_\_ ?

- (a)  $t=3$
- (b)  $t=5$
- (c)  $t=8$
- (d)  $t=15$
- (e) none of the above

20. (2 pts) What goes in the blank  $e=$ \_\_\_  $f=$ \_\_\_ ?

- (a)  $e=5$   $f=4$
- (b)  $e=4$   $f=5$
- (c)  $e=5$   $f=5$
- (d)  $e=4$   $f=4$
- (e) none of the above

21. (20 pts) The main program listed below demonstrates a function that can put an array of temperature values in order from hottest to coldest. Sample output is given, along with a complete main program, and three function definitions. The last two function definitions on the following page are incomplete. Finish them, using the hints provided. Be extra careful with the order of parameters.

Sample output:

```
> ./a.out
Temps: 45.0 32.0 54.0 8.5 19.1
Hottest to Coldest: 54.0 45.0 32.0 19.1 8.5
>
```

Code:

```
/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */
/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}
```

```

void outputArrayOnOneLine(int count, double array[])
{
 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?

 About 3-4 lines of code is all you need.
 */

}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array

 About 6-8 lines of code will be enough. */

 int j;
 double temp;
 int indexOfColdest = 0; /* index of coldest element */

}

```

The following questions deal with this code excerpt:

```
// quiz question
#include <iostream>
using namespace std;

struct Point_S
{
 int x;
 int y;
};

int main(int argc, char*argv[])

{
 int a;
 int *b;
 int *c;

 double d;
 double e;
 double *f;
 double *g;

 Point_S *p;
 Point_S *q;
 Point_S *r;
 Point_S s;
 Point_S t;

 b = new int;
 c = &a;

 f = new double;
 g = &e;

 p = new Point_S;
 q = new Point_S;
 r = &s;

 cout << "Hi" << endl;
 return 0;
}
```

22. (40 pts) Suppose you run the C++ program on the previous page, and the program reaches the line of code that prints out Hi.

Indicate whether, at that point in time, the expression given in the table on the left

- refers to a memory location on the *stack*,
- refers to a memory location on the *heap*,
- would cause an *error* (e.g. dereferencing something that isn't a pointer.)

Then, in the table on the right, fill in the type of each expression, or write "error" if the expression would not be valid (e.g. dereferencing something that isn't a pointer, or using the dot operator (.) on something that isn't a struct.

| <i>expression</i> | <i>Circle stack, heap or error</i> |
|-------------------|------------------------------------|
| a                 | stack heap error                   |
| *a                | stack heap error                   |
| b                 | stack heap error                   |
| *b                | stack heap error                   |
| c                 | stack heap error                   |
| *c                | stack heap error                   |
| *d                | stack heap error                   |
| *e                | stack heap error                   |
| f                 | stack heap error                   |
| *f                | stack heap error                   |
| g                 | stack heap error                   |
| *g                | stack heap error                   |
| p                 | stack heap error                   |
| *p                | stack heap error                   |
| *q                | stack heap error                   |
| *r                | stack heap error                   |
| *s                | stack heap error                   |
| *t                | stack heap error                   |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int         |
| *a                |             |
| &a                |             |
| b                 |             |
| *b                |             |
| &b                |             |
| d                 |             |
| *d                |             |
| e                 |             |
| &e                |             |
| p                 |             |
| *r                |             |
| &s                |             |
| t->x              |             |
| q.y               |             |
| q->y              |             |
| (*p).x            |             |
| &(p.x)            |             |
| argc              |             |
| argv[0]           |             |
| argv[0][1]        |             |
| argv              |             |
| &argc             |             |

End of Exam. Total Points: 100

XYAFRA91BDPE38A82D6DF3S

**A** KEY CISC 181 sections 010-012, Midterm 2

11/11/05

Questions 1 through 4 deal with the following code excerpt:

```
1 // foo.cc
2
3 #include <iostream>
4 using namespace std;
5
6 int main(void)
7 {
8 char mascot[10]="Blue Hens";
9 char hometown[20]="Newark";
10 char school[30]="University of Delaware";
11 char state[20]="Delaware";
12
13 char *a = mascot;
14 char * const c = hometown;
15 const char *b = school;
16 const char * const d = state;
17
18
19 cout << a << endl;
20 cout << b << endl;
21 cout << c << endl;
22 cout << d << endl;
23
24 // where some extra code might go
25
26 #include "extra.h"
27
28 cout << a << endl;
29 cout << b << endl;
30 cout << c << endl;
31 cout << d << endl;
32
33 }
34
35
36
37
38
39
40
41
42
43
44
45
46
```

1. (2 pts) One of these lines of code would produce a compiler error if placed at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a[0] = 'F';`
- (b) `b = hometown;`
- (c) `a[1] = 'r';`
- (d) `b[0] = 'D';`
- (e) `a = hometown;`

2. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a = mascot;`
- (b) `a = state;`
- (c) `b[1] = 'u';`
- (d) `c[1] = 'm';`
- (e) `a[1] = 'u';`

3. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `d = state;`
- (b) `b = hometown;`
- (c) `c[0] = 'D';`
- (d) `b = mascot`
- (e) `c[1] = 'i';`

4. (2 pts) Read carefully! Only one of these lines of code would NOT produce a compiler error if placed at the spot indicated by // where some extra code might go ?

That is, which one is CORRECT?

- (a) `c[1] = 'u';`
- (b) `a[1] = 'r';`
- (c) `c = d;`
- (d) `c = a;`
- (e) `d = hometown;`

Questions 5 through 8 refer to the fragment of code shown in Figure 1 that comes from a complete C++ program.

Figure 2: Declaration of a struct

```
...
struct Node
{
 int data;
 Node *next;
};

int main(void)
{
 Node *p;
 Node s;
 p = new Node;
 ...
}
```

5. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 1, at or after the line indicated by ...  
Which one?
- (a) data = 3
  - (b) s.data = 3;
  - (c) (\*s).data = 3;
  - (d) s->data = 3;
  - (e) p.data = 3;
6. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 1, at or after the line indicated by ...  
Which one?
- (a) next = NULL;
  - (b) p->next = NULL;
  - (c) s->next = NULL;
  - (d) p.next = NULL;
  - (e) (\*s).next = NULL;
7. (2 pts) Which of the following is equivalent to p->data?
- (a) (\*p).data
  - (b) \*(p.data)
  - (c) p.(\*data)
  - (d) p[data]
  - (e) p[1].data

8. (2 pts) Which of the following is equivalent to `p->next`? (Hint: This is is tricky: think carefully about the equivalence of array subscripts and dereferencing of pointers.)

- (a) `*(p.next)`
- (b) `p[next]`
- (c) `p[0].next`
- (d) `p[1].next`
- (e) `*p[next]`

9. (2 pts) Suppose you want to write a function `incrModFive` that will increment an integer mod 5 (i.e. if the integer is 4 and you increment it, it will return to 0.) If you want the function `incrModFive` to take one parameter, an `int` passed by reference, and return nothing at all, which of the following would be a correct function prototype for `incrModFive`?

- (a) `incrModFive(int x);`
- (b) `void incrModFive(int x);`
- (c) `void incrModFive(int &x);`
- (d) `int incrModFive(int x);`
- (e) `incrModFive (int &x);`

10. (2 pts) (Continuation of previous question.) If you then called that function, which of the following is a legal call to function `incrModFive`? Assume that the following declarations are in effect:

```
int y, z;
y = 3;
z = 4;
int *x;
x = &y;
```

- (a) `incrModFive(&z);`
- (b) `incrModFive(y);`
- (c) `incrModFive(x);`
- (d) `incrModFive(*y);`
- (e) `z=incrModFive(4);`

11. (2 pts) If the variable `a` refers to an array of 10 elements, then the expression `a[0]` is equivalent to which of the following?

- (a) `a`
- (b) `*a`
- (c) `a+0`
- (d) `a++`
- (e) `++a`

12. (2 pts) Suppose two function definitions in the same program have the same name, but differ in the number and/or type of their formal parameters, as illustrated by these two function prototypes:

```
int max(int a, int b);
int max(int a, int b, int c);
```

This would be an example of which of the following?

- (a) a binary scope resolution operator
  - (b) function overloading
  - (c) a unary scope resolution operator
  - (d) separate compilation
  - (e) a syntax error, since this is not permitted
13. (2 pts) If you want the variable `a` to contain the address of an `double`, which of the following would be a correct declaration of `a`?
- (a) `double a;`
  - (b) `double &a;`
  - (c) `double *a;`
  - (d) `double %a;`
  - (e) none of the above
14. (2 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?
- (a) `*p`
  - (b) `&p`
  - (c) `deref(p)`
  - (d) `%p`
  - (e) `$p`
15. (2 pts) Suppose a UD student named Terry Bly Smart has the userid `tbsmart`. Which of the following URLs brings up his/her personal web page on `strauss`?
- (Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)
- (a) `http://strauss.udel.edu/tbsmart`
  - (b) `http://strauss.udel.edu/~tbsmart`
  - (c) `http://strauss.udel.edu/~tbsmart/public.html`
  - (d) `http://strauss.udel.edu/~tbsmart/public.html`

16. (2 pts) Suppose in a future semester, your instructor for CISC321 requires you to create a web page on strauss with the URL `http://strauss.udel.edu/~userid/CISC321`. Which of the following files could you edit to create the web page?

(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)

- (a) `~/CISC321/index.html`
- (b) `~/public_html/CISC321/index.html`
- (c) `~/public_html/cisc321/index.html`
- (d) `~/public_html/CISC321/index.html`

Here is a sample program, followed by some sample output.—except, the numbers have been replaced with blanks First, trace through the program to determine what goes in each of these blanks, then select the correct answers from the choices given in the problems.

**Don't panic:** Each of the sections of code can be done independently of the other sections, so if you are stuck on one, don't stop; keep working.

```
1 /* q10.cc Agnes Nitt for CISC181 final exam */
2
3 #include <iostream>
4 using namespace std;
5
6 void trouble(int *r);
7 double fire(double f);
8 int burn(int b, int *n);
9 void cauldron(int &c, int &d);
10
11 int main(void)
12 {
13 int e=2, f=3, t = 4, y = 9;
14 double v=5.1, z = 10.2;
15
16 trouble(&y);
17 cout << "y=" << y << endl;
18
19 z = fire(v);
20 cout << "v=" << v << " z=" << z << endl;
21
22 t = 3;
23 burn(5, &t);
24 cout << "t=" << t << endl;
25
26 e=4;
27 f=5;
28 cauldron(e,f);
29 cout << "e=" << e << " f=" << f << endl;
30 return 0;
31 }
32
33 void trouble(int *r)
34 {
35 (*r) = -1;
36 return;
37 }
38
39 void cauldron(int &c, int &d)
40 {
41 int temp = c;
```

```
42 | c = d;
43 | d = temp;
44 | }
45 |
46 | int burn(int b, int *n)
47 | {
48 | (*n) = b;
49 | }
50 |
51 | double fire(double f)
52 | {
53 | return (f + 0.1);
54 | }
```

```
> ./q10
y=____
v=____ z=____
t=____
e=____ f=____
>
```

17. (2 pts) What goes in the blank  $y=$ \_\_\_ ?

- (a)  $y=-1$
- (b)  $y=-9$
- (c)  $y=9$
- (d)  $y=-8$
- (e) none of the above

18. (2 pts) What goes in the blank  $v=$ \_\_\_  $z=$ \_\_\_ ?

- (a)  $v=5.1$   $z=5.1$
- (b)  $v=5.2$   $z=5.2$
- (c)  $v=5.2$   $z=5.1$
- (d)  $v=5.1$   $z=5.2$
- (e) none of the above

19. (2 pts) What goes in the blank  $t=$ \_\_\_ ?

- (a)  $t=3$
- (b)  $t=5$
- (c)  $t=8$
- (d)  $t=15$
- (e) none of the above

20. (2 pts) What goes in the blank  $e=$ \_\_\_  $f=$ \_\_\_ ?

- (a)  $e=5$   $f=4$
- (b)  $e=4$   $f=5$
- (c)  $e=5$   $f=5$
- (d)  $e=4$   $f=4$
- (e) none of the above

21. (20 pts) The main program listed below demonstrates a function that can put an array of temperature values in order from hottest to coldest. Sample output is given, along with a complete main program, and three function definitions. The last two function definitions on the following page are incomplete. Finish them, using the hints provided. Be extra careful with the order of parameters.

Sample output:

```
> ./a.out
Temps: 45.0 32.0 54.0 8.5 19.1
Hottest to Coldest: 54.0 45.0 32.0 19.1 8.5
>
```

Code:

```
/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */
/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}

void outputArrayOnOneLine(int count, double array[])
{
```

```

 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?

 About 3-4 lines of code is all you need.
 */
}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array

 About 6-8 lines of code will be enough. */

 int j;
 double temp;
 int indexOfColdest = 0; /* index of coldest element */

}

/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */

```

```

/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}

void outputArrayOnOneLine(int count, double array[])
{
 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?
 */

 for (int i=howMany; i>1; i--)
 {
 putColdestAtEnd(temps, i);
 }
}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array */

```

```

int j;
double temp;
int indexOfColdest = 0; /* index of coldest element */

for (j=1; j<numTemps; j++)
{
 if (tempArray[j] < tempArray[indexOfColdest])
 {
 indexOfColdest = j;
 }
}

temp = tempArray[indexOfColdest];
tempArray[indexOfColdest] = tempArray[numTemps - 1];
tempArray[numTemps - 1] = temp;
}

```

The following questions deal with this code excerpt:

```

// quiz question
#include <iostream>
using namespace std;

struct Point_S
{
 int x;
 int y;
};

int main(int argc, char*argv[])

{
 int a;
 int *b;
 int *c;

 double d;
 double e;
 double *f;
 double *g;

 Point_S *p;
 Point_S *q;
 Point_S *r;
 Point_S s;
 Point_S t;
}

```

```
b = new int;
c = &a;

f = new double;
g = &e;

p = new Point_S;
q = new Point_S;
r = &s;

cout << "Hi" << endl;
return 0;
}
```

22. (40 pts) Suppose you run the C++ program on the previous page, and the program reaches the line of code that prints out `Hi`.

Indicate whether, at that point in time, the expression given in the table on the left

- refers to a memory location on the *stack*,
- refers to a memory location on the *heap*,
- would cause an *error* (e.g. dereferencing something that isn't a pointer.)

Then, in the table on the right, fill in the type of each expression, or write "error" if the expression would not be valid (e.g. dereferencing something that isn't a pointer, or using the dot operator (`.`) on something that isn't a struct).

| <i>expression</i> | <i>Circle stack, heap or error</i> |
|-------------------|------------------------------------|
| a                 | stack heap error                   |
| *a                | stack heap error                   |
| b                 | stack heap error                   |
| *b                | stack heap error                   |
| c                 | stack heap error                   |
| *c                | stack heap error                   |
| *d                | stack heap error                   |
| *e                | stack heap error                   |
| f                 | stack heap error                   |
| *f                | stack heap error                   |
| g                 | stack heap error                   |
| *g                | stack heap error                   |
| p                 | stack heap error                   |
| *p                | stack heap error                   |
| *q                | stack heap error                   |
| *r                | stack heap error                   |
| *s                | stack heap error                   |
| *t                | stack heap error                   |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int         |
| *a                |             |
| &a                |             |
| b                 |             |
| *b                |             |
| &b                |             |
| d                 |             |
| *d                |             |
| e                 |             |
| &e                |             |
| p                 |             |
| *r                |             |
| &s                |             |
| t->x              |             |
| q.y               |             |
| q->y              |             |
| (*p).x            |             |
| &(p.x)            |             |
| argc              |             |
| argv[0]           |             |
| argv[0][1]        |             |
| argv              |             |
| &argc             |             |

| <i>expression</i> | <i>Answer</i> |
|-------------------|---------------|
| a                 | stack         |
| *a                | error         |
| b                 | stack         |
| *b                | heap          |
| c                 | stack         |
| *c                | stack         |
| *d                | error         |
| *e                | error         |
| f                 | stack         |
| *f                | heap          |
| g                 | stack         |
| *g                | stack         |
| p                 | stack         |
| *p                | heap          |
| *q                | heap          |
| *r                | stack         |
| *s                | error         |
| *t                | error         |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int         |
| *a                | error       |
| &a                | int *       |
| b                 | int*        |
| *b                | int         |
| &b                | int**       |
| d                 | double      |
| *d                | error       |
| e                 | double      |
| &e                | double *    |
| p                 | Point_S *   |
| *r                | Point_S     |
| &s                | Point_S *   |
| t->x              | error       |
| q.y               | error       |
| q->y              | int         |
| (*p).x            | int         |
| &(p.x)            | error       |
| argc              | int         |
| argv[0]           | char *      |
| argv[0][1]        | char        |
| argv              | char **     |
| &argc             | int *       |

End of Key, seed 3456 version **A**  
Total Points: 100

Name \_\_\_\_\_

Please circle your section number:

010 (Wed PM)

011 (Mon PM)

012 (Mon AM)

Answer the multiple choice questions on a “Scantron Form”

Bubble in **ONLY** your Unix userid and your answers

**DO NOT** bubble in your id number or section

If you bubble in your SSN, the computer will **reject your form!!!**

Answer the remaining questions directly on the exam paper.

### General Instructions

- The exam is @@@% multiple choice, and @@@% programming.
- The programming questions start with number 21. You may want to tackle them first, since they may take more time.
- **DO NOT WRITE YOUR NAME ON ANY PAGE EXCEPT THIS ONE!**
- You have 50 minutes. **Pace yourself**, and pay attention to the point values.
- Read *all* the directions *carefully* on each problem.
- Good luck.

Questions 1 through 4 deal with the following code excerpt:

```
1 // foo.cc
2
3 #include <iostream>
4 using namespace std;
5
6 int main(void)
7 {
8 char mascot[10]="Blue Hens";
9 char hometown[20]="Newark";
10 char school[30]="University of Delaware";
11 char state[20]="Delaware";
12
13 char *a = mascot;
14 char * const c = hometown;
15 const char *b = school;
16 const char * const d = state;
17
18
19 cout << a << endl;
20 cout << b << endl;
21 cout << c << endl;
22 cout << d << endl;
23
24 // where some extra code might go
25
26 #include "extra.h"
27
28 cout << a << endl;
29 cout << b << endl;
30 cout << c << endl;
31 cout << d << endl;
32
33 }
34
35
36
37
38
39
40
41
42
43
44
45
46
```

1. (2 pts) One of these lines of code would produce a compiler error if placed at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a[0] = 'F';`
- (b) `b = hometown;`
- (c) `a[1] = 'r';`
- (d) `b[0] = 'D';`
- (e) `a = hometown;`

2. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a = mascot;`
- (b) `a = state;`
- (c) `b[1] = 'u';`
- (d) `c[1] = 'm';`
- (e) `a[1] = 'u';`

3. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `d = state;`
- (b) `b = hometown;`
- (c) `c[0] = 'D';`
- (d) `b = mascot`
- (e) `c[1] = 'i';`

4. (2 pts) Read carefully! Only one of these lines of code would NOT produce a compiler error if placed at the spot indicated by // where some extra code might go ?

That is, which one is CORRECT?

- (a) `c[1] = 'u';`
- (b) `a[1] = 'r';`
- (c) `c = d;`
- (d) `c = a;`
- (e) `d = hometown;`

Questions 5 through 8 refer to the fragment of code shown in Figure 3 that comes from a complete C++ program.

Figure 3: Declaration of a struct

```
...
struct Node
{
 int data;
 Node *next;
};

int main(void)
{
 Node *p;
 Node s;
 p = new Node;
 ...
}
```

5. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 3, at or after the line indicated by . . .  
Which one?
- (a) `data = 3`
  - (b) `s.data = 3;`
  - (c) `(*s).data = 3;`
  - (d) `s->data = 3;`
  - (e) `p.data = 3;`
6. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 3, at or after the line indicated by . . .  
Which one?
- (a) `next = NULL;`
  - (b) `p->next = NULL;`
  - (c) `s->next = NULL;`
  - (d) `p.next = NULL;`
  - (e) `(*s).next = NULL;`

7. (2 pts) Which of the following is equivalent to `p->data`?
- (a) `(*p).data`
  - (b) `*(p.data)`
  - (c) `p.(*data)`
  - (d) `p[data]`
  - (e) `p[1].data`
8. (2 pts) Which of the following is equivalent to `p->next`? (Hint: This is is tricky: think carefully about the equivalence of array subscripts and dereferencing of pointers.)
- (a) `*(p.next)`
  - (b) `p[next]`
  - (c) `p[0].next`
  - (d) `p[1].next`
  - (e) `*p[next]`
9. (2 pts) If you want the variable `a` to contain the address of an `double`, which of the following would be a correct declaration of `a`?
- (a) `double a;`
  - (b) `double &a;`
  - (c) `double *a;`
  - (d) `double %a;`
  - (e) none of the above
10. (2 pts) Suppose two function definitions in the same program have the same name, but differ in the number and/or type of their formal parameters, as illustrated by these two function prototypes:

```
int max(int a, int b);
int max(int a, int b, int c);
```

This would be an example of which of the following?

- (a) a binary scope resolution operator
- (b) function overloading
- (c) a unary scope resolution operator
- (d) separate compilation
- (e) a syntax error, since this is not permitted

11. (2 pts) If the variable `a` refers to an array of 10 elements, then the expression `a[0]` is equivalent to which of the following?
- (a) `a`
  - (b) `*a`
  - (c) `a+0`
  - (d) `a++`
  - (e) `++a`
12. (2 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?
- (a) `*p`
  - (b) `&p`
  - (c) `deref(p)`
  - (d) `%p`
  - (e) `$p`
13. (2 pts) Suppose you want to write a function `incrModFive` that will increment an integer mod 5 (i.e. if the integer is 4 and you increment it, it will return to 0.) If you want the function `incrModFive` to take one parameter, an `int` passed by reference, and return nothing at all, which of the following would be a correct function prototype for `incrModFive`?
- (a) `incrModFive(int x);`
  - (b) `void incrModFive(int x);`
  - (c) `void incrModFive(int &x);`
  - (d) `int incrModFive(int x);`
  - (e) `incrModFive (int &x);`
14. (2 pts) (Continuation of previous question.) If you then called that function, which of the following is a legal call to function `incrModFive`? Assume that the following declarations are in effect:

```
int y,z;
y = 3;
z = 4;
int *x;
x = &y;
```

- (a) `incrModFive(&z);`
- (b) `incrModFive(y);`
- (c) `incrModFive(x);`
- (d) `incrModFive(*y);`
- (e) `z=incrModFive(4);`

15. (2 pts) Suppose a UD student named Terry Bly Smart has the userid tbsmart. Which of the following URLs brings up his/her personal web page on strauss?  
(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)
- (a) `http://strauss.udel.edu/tbsmart`
  - (b) `http://strauss.udel.edu/~tbsmart`
  - (c) `http://strauss.udel.edu/~tbsmart/public.html`
  - (d) `http://strauss.udel.edu/~tbsmart/public.html`
16. (2 pts) Suppose in a future semester, your instructor for CISC321 requires you to create a web page on strauss with the URL `http://strauss.udel.edu/~userid/CISC321`. Which of the following files could you edit to create the web page?  
(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)
- (a) `~/CISC321/index.html`
  - (b) `~/public_html/CISC321/index.html`
  - (c) `~/public_html/cisc321/index.html`
  - (d) `~/public_html/CISC321/index.html`

Here is a sample program, followed by some sample output.—except, the numbers have been replaced with blanks First, trace through the program to determine what goes in each of these blanks, then select the correct answers from the choices given in the problems.

**Don't panic:** Each of the sections of code can be done independently of the other sections, so if you are stuck on one, don't stop; keep working.

```
1 /* ql0.cc Agnes Nitt for CISC181 final exam */
2
3 #include <iostream>
4 using namespace std;
5
6 void trouble(int *r);
7 double fire(double f);
8 int burn(int b, int *n);
9 void cauldron(int &c, int &d);
10
11 int main(void)
12 {
13 int e=2, f=3, t = 4, y = 9;
14 double v=5.1, z = 10.2;
15
16 trouble(&y);
17 cout << "y=" << y << endl;
18
19 z = fire(v);
20 cout << "v=" << v << " z=" << z << endl;
21
22 t = 3;
23 burn(5, &t);
24 cout << "t=" << t << endl;
25
26 e=4;
27 f=5;
28 cauldron(e,f);
29 cout << "e=" << e << " f=" << f << endl;
30 return 0;
31 }
32
33 void trouble(int *r)
34 {
35 (*r) = -1;
36 return;
37 }
38
39 void cauldron(int &c, int &d)
40 {
41 int temp = c;
42 c = d;
43 d = temp;
44 }
45
46 int burn(int b, int *n)
47 {
48 (*n) = b;
49 }
50
51 double fire(double f)
52 {
53 return (f + 0.1);
54 }
```

```
> ./q10
y=___
v=___ z=___
t=___
e=___ f=___
>
```

17. (2 pts) What goes in the blank  $y=$ \_\_\_ ?

- (a)  $y=-1$
- (b)  $y=-9$
- (c)  $y=9$
- (d)  $y=-8$
- (e) none of the above

18. (2 pts) What goes in the blank  $v=$ \_\_\_  $z=$ \_\_\_ ?

- (a)  $v=5.1$   $z=5.1$
- (b)  $v=5.2$   $z=5.2$
- (c)  $v=5.2$   $z=5.1$
- (d)  $v=5.1$   $z=5.2$
- (e) none of the above

19. (2 pts) What goes in the blank  $t=$ \_\_\_ ?

- (a)  $t=3$
- (b)  $t=5$
- (c)  $t=8$
- (d)  $t=15$
- (e) none of the above

20. (2 pts) What goes in the blank  $e=$ \_\_\_  $f=$ \_\_\_ ?

- (a)  $e=5$   $f=4$
- (b)  $e=4$   $f=5$
- (c)  $e=5$   $f=5$
- (d)  $e=4$   $f=4$
- (e) none of the above

21. (20 pts) The main program listed below demonstrates a function that can put an array of temperature values in order from hottest to coldest. Sample output is given, along with a complete main program, and three function definitions. The last two function definitions on the following page are incomplete. Finish them, using the hints provided. Be extra careful with the order of parameters.

Sample output:

```
> ./a.out
Temps: 45.0 32.0 54.0 8.5 19.1
Hottest to Coldest: 54.0 45.0 32.0 19.1 8.5
>
```

Code:

```
/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */
/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}
```

```

void outputArrayOnOneLine(int count, double array[])
{
 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?

 About 3-4 lines of code is all you need.
 */

}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array

 About 6-8 lines of code will be enough. */

 int j;
 double temp;
 int indexOfColdest = 0; /* index of coldest element */

}

```

The following questions deal with this code excerpt:

```
// quiz question
#include <iostream>
using namespace std;

struct Point_S
{
 int x;
 int y;
};

int main(int argc, char*argv[])

{
 int *a;
 int *b;
 int c;

 double d;
 double *e;
 double *f;
 double g;

 Point_S p;
 Point_S *q;
 Point_S *r;
 Point_S *s;
 Point_S t;

 a = new int;
 b=&c;

 e = new double;
 f = &d;

 q = new Point_S;
 r = &p ;
 s = &t;

 cout << "Hi" << endl;
 return 0;
}
```

22. (40 pts) Suppose you run the C++ program on the previous page, and the program reaches the line of code that prints out Hi.

Indicate whether, at that point in time, the expression given in the table on the left

- refers to a memory location on the *stack*,
- refers to a memory location on the *heap*,
- would cause an *error* (e.g. dereferencing something that isn't a pointer.)

Then, in the table on the right, fill in the type of each expression, or write "error" if the expression would not be valid (e.g. dereferencing something that isn't a pointer, or using the dot operator (.) on something that isn't a struct.

| <i>expression</i> | <i>Circle stack, heap or error</i> |
|-------------------|------------------------------------|
| a                 | stack heap error                   |
| *a                | stack heap error                   |
| b                 | stack heap error                   |
| *b                | stack heap error                   |
| c                 | stack heap error                   |
| *c                | stack heap error                   |
| *d                | stack heap error                   |
| *e                | stack heap error                   |
| f                 | stack heap error                   |
| *f                | stack heap error                   |
| g                 | stack heap error                   |
| *g                | stack heap error                   |
| p                 | stack heap error                   |
| *p                | stack heap error                   |
| *q                | stack heap error                   |
| *r                | stack heap error                   |
| *s                | stack heap error                   |
| *t                | stack heap error                   |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int *       |
| *a                |             |
| &a                |             |
| b                 |             |
| *b                |             |
| &b                |             |
| d                 |             |
| *d                |             |
| e                 |             |
| &e                |             |
| p                 |             |
| *r                |             |
| &s                |             |
| t->x              |             |
| q.Y               |             |
| q->y              |             |
| (*p).x            |             |
| &(p.x)            |             |
| argc              |             |
| argv[0]           |             |
| argv[0][1]        |             |
| argv              |             |
| &argc             |             |

End of Exam. Total Points: 100

XYBFRB92CEPE38C82D6CF3T

**B** KEY CISC 181 sections 010-012, Midterm 2

11/11/05

Questions 1 through 4 deal with the following code excerpt:

```
1 // foo.cc
2
3 #include <iostream>
4 using namespace std;
5
6 int main(void)
7 {
8 char mascot[10]="Blue Hens";
9 char hometown[20]="Newark";
10 char school[30]="University of Delaware";
11 char state[20]="Delaware";
12
13 char *a = mascot;
14 char * const c = hometown;
15 const char *b = school;
16 const char * const d = state;
17
18
19 cout << a << endl;
20 cout << b << endl;
21 cout << c << endl;
22 cout << d << endl;
23
24 // where some extra code might go
25
26 #include "extra.h"
27
28 cout << a << endl;
29 cout << b << endl;
30 cout << c << endl;
31 cout << d << endl;
32
33 }
34
35
36
37
38
39
40
41
42
43
44
45
46
```

1. (2 pts) One of these lines of code would produce a compiler error if placed at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a[0] = 'F';`
- (b) `b = hometown;`
- (c) `a[1] = 'r';`
- (d) `b[0] = 'D';`
- (e) `a = hometown;`

2. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a = mascot;`
- (b) `a = state;`
- (c) `b[1] = 'u';`
- (d) `c[1] = 'm';`
- (e) `a[1] = 'u';`

3. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `d = state;`
- (b) `b = hometown;`
- (c) `c[0] = 'D';`
- (d) `b = mascot`
- (e) `c[1] = 'i';`

4. (2 pts) Read carefully! Only one of these lines of code would NOT produce a compiler error if placed at the spot indicated by // where some extra code might go ?

That is, which one is CORRECT?

- (a) `c[1] = 'u';`
- (b) `a[1] = 'r';`
- (c) `c = d;`
- (d) `c = a;`
- (e) `d = hometown;`

Questions 5 through 8 refer to the fragment of code shown in Figure 3 that comes from a complete C++ program.

Figure 4: Declaration of a struct

```
...
struct Node
{
 int data;
 Node *next;
};

int main(void)
{
 Node *p;
 Node s;
 p = new Node;
 ...
}
```

5. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 3, at or after the line indicated by ...  
Which one?
- (a) data = 3
  - (b) s.data = 3;
  - (c) (\*s).data = 3;
  - (d) s->data = 3;
  - (e) p.data = 3;
6. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 3, at or after the line indicated by ...  
Which one?
- (a) next = NULL;
  - (b) p->next = NULL;
  - (c) s->next = NULL;
  - (d) p.next = NULL;
  - (e) (\*s).next = NULL;
7. (2 pts) Which of the following is equivalent to p->data?
- (a) (\*p).data
  - (b) \*(p.data)
  - (c) p.(\*data)
  - (d) p[data]
  - (e) p[1].data

8. (2 pts) Which of the following is equivalent to `p->next`? (Hint: This is tricky: think carefully about the equivalence of array subscripts and dereferencing of pointers.)

- (a) `*(p.next)`
- (b) `p[next]`
- (c) `p[0].next`
- (d) `p[1].next`
- (e) `*p[next]`

9. (2 pts) If you want the variable `a` to contain the address of an `double`, which of the following would be a correct declaration of `a`?

- (a) `double a;`
- (b) `double &a;`
- (c) `double *a;`
- (d) `double %a;`
- (e) none of the above

10. (2 pts) Suppose two function definitions in the same program have the same name, but differ in the number and/or type of their formal parameters, as illustrated by these two function prototypes:

```
int max(int a, int b);
int max(int a, int b, int c);
```

This would be an example of which of the following?

- (a) a binary scope resolution operator
- (b) function overloading
- (c) a unary scope resolution operator
- (d) separate compilation
- (e) a syntax error, since this is not permitted

11. (2 pts) If the variable `a` refers to an array of 10 elements, then the expression `a[0]` is equivalent to which of the following?

- (a) `a`
- (b) `*a`
- (c) `a+0`
- (d) `a++`
- (e) `++a`

12. (2 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?

- (a) `*p`
- (b) `&p`
- (c) `deref(p)`
- (d) `%p`
- (e) `$p`

13. (2 pts) Suppose you want to write a function `incrModFive` that will increment an integer mod 5 (i.e. if the integer is 4 and you increment it, it will return to 0.) If you want the function `incrModFive` to take one parameter, an `int` passed by reference, and return nothing at all, which of the following would be a correct function prototype for `incrModFive`?

(a) `incrModFive(int x);`  
(b) `void incrModFive(int x);`  
 (c) `void incrModFive(int &x);`  
(d) `int incrModFive(int x);`  
(e) `incrModFive (int &x);`

14. (2 pts) (Continuation of previous question.) If you then called that function, which of the following is a legal call to function `incrModFive`? Assume that the following declarations are in effect:

```
int y, z;
y = 3;
z = 4;
int *x;
x = &y;
```

(a) `incrModFive(&z);`  
 (b) `incrModFive(y);`  
(c) `incrModFive(x);`  
(d) `incrModFive(*y);`  
(e) `z=incrModFive(4);`

15. (2 pts) Suppose a UD student named Terry Bly Smart has the userid `tbsmart`. Which of the following URLs brings up his/her personal web page on `strauss`?

(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)

(a) `http://strauss.udel.edu/tbsmart`  
 (b) `http://strauss.udel.edu/~tbsmart`  
(c) `http://strauss.udel.edu/~tbsmart/public.html`  
(d) `http://strauss.udel.edu/~tbsmart/public.html`

16. (2 pts) Suppose in a future semester, your instructor for CISC321 requires you to create a web page on `strauss` with the URL

`http://strauss.udel.edu/~userid/CISC321`. Which of the following files could you edit to create the web page?

(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)

(a) `~/CISC321/index.html`  
(b) `~/public_html/CISC321/index.html`  
(c) `~/public_html/cisc321/index.html`  
 (d) `~/public_html/CISC321/index.html`

Here is a sample program, followed by some sample output.—except, the numbers have been replaced with blanks First, trace through the program to determine what goes in each of these blanks, then select the correct answers from the choices given in the problems.

**Don't panic:** Each of the sections of code can be done independently of the other sections, so if you are stuck on one, don't stop; keep working.

```
1 /* q10.cc Agnes Nitt for CISC181 final exam */
2
3 #include <iostream>
4 using namespace std;
5
6 void trouble(int *r);
7 double fire(double f);
8 int burn(int b, int *n);
9 void cauldron(int &c, int &d);
10
11 int main(void)
12 {
13 int e=2, f=3, t = 4, y = 9;
14 double v=5.1, z = 10.2;
15
16 trouble(&y);
17 cout << "y=" << y << endl;
18
19 z = fire(v);
20 cout << "v=" << v << " z=" << z << endl;
21
22 t = 3;
23 burn(5, &t);
24 cout << "t=" << t << endl;
25
26 e=4;
27 f=5;
28 cauldron(e,f);
29 cout << "e=" << e << " f=" << f << endl;
30 return 0;
31 }
32
33 void trouble(int *r)
34 {
35 (*r) = -1;
36 return;
37 }
38
39 void cauldron(int &c, int &d)
40 {
41 int temp = c;
42 c = d;
43 d = temp;
44 }
45
46 int burn(int b, int *n)
47 {
48 (*n) = b;
49 }
50
51 double fire(double f)
52 {
53 return (f + 0.1);
54 }
```

> ./q10

y=\_\_\_\_

v=\_\_\_\_ z=\_\_\_\_

t=\_\_\_\_  
e=\_\_\_\_ f=\_\_\_\_  
>

17. (2 pts) What goes in the blank  $y=$ \_\_\_ ?

- (a)  $y=-1$
- (b)  $y=-9$
- (c)  $y=9$
- (d)  $y=-8$
- (e) none of the above

18. (2 pts) What goes in the blank  $v=$ \_\_\_  $z=$ \_\_\_ ?

- (a)  $v=5.1$   $z=5.1$
- (b)  $v=5.2$   $z=5.2$
- (c)  $v=5.2$   $z=5.1$
- (d)  $v=5.1$   $z=5.2$
- (e) none of the above

19. (2 pts) What goes in the blank  $t=$ \_\_\_ ?

- (a)  $t=3$
- (b)  $t=5$
- (c)  $t=8$
- (d)  $t=15$
- (e) none of the above

20. (2 pts) What goes in the blank  $e=$ \_\_\_  $f=$ \_\_\_ ?

- (a)  $e=5$   $f=4$
- (b)  $e=4$   $f=5$
- (c)  $e=5$   $f=5$
- (d)  $e=4$   $f=4$
- (e) none of the above

21. (20 pts) The main program listed below demonstrates a function that can put an array of temperature values in order from hottest to coldest. Sample output is given, along with a complete main program, and three function definitions. The last two function definitions on the following page are incomplete. Finish them, using the hints provided. Be extra careful with the order of parameters.

Sample output:

```
> ./a.out
Temps: 45.0 32.0 54.0 8.5 19.1
Hottest to Coldest: 54.0 45.0 32.0 19.1 8.5
>
```

Code:

```
/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */
/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}

void outputArrayOnOneLine(int count, double array[])
{
```

```

 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?

 About 3-4 lines of code is all you need.
 */

}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array

 About 6-8 lines of code will be enough. */

 int j;
 double temp;
 int indexOfColdest = 0; /* index of coldest element */

}

/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */

```

```

/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}

void outputArrayOnOneLine(int count, double array[])
{
 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?
 */

 for (int i=howMany; i>1; i--)
 {
 putColdestAtEnd(temps, i);
 }
}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array */

```

```

int j;
double temp;
int indexOfColdest = 0; /* index of coldest element */

for (j=1; j<numTemps; j++)
{
 if (tempArray[j] < tempArray[indexOfColdest])
 {
 indexOfColdest = j;
 }
}

temp = tempArray[indexOfColdest];
tempArray[indexOfColdest] = tempArray[numTemps - 1];
tempArray[numTemps - 1] = temp;
}

```

The following questions deal with this code excerpt:

```

// quiz question
#include <iostream>
using namespace std;

struct Point_S
{
 int x;
 int y;
};

int main(int argc, char*argv[])

{
 int *a;
 int *b;
 int c;

 double d;
 double *e;
 double *f;
 double g;

 Point_S p;
 Point_S *q;
 Point_S *r;
 Point_S *s;
 Point_S t;
}

```

```

a = new int;
b=&c;

e = new double;
f = &d;

q = new Point_S;
r = &p ;
s = &t;

cout << "Hi" << endl;
return 0;
}

```

22. (40 pts) Suppose you run the C++ program on the previous page, and the program reaches the line of code that prints out `Hi`.

Indicate whether, at that point in time, the expression given in the table on the left

- refers to a memory location on the *stack*,
- refers to a memory location on the *heap*,
- would cause an *error* (e.g. dereferencing something that isn't a pointer.)

Then, in the table on the right, fill in the type of each expression, or write "error" if the expression would not be valid (e.g. dereferencing something that isn't a pointer, or using the dot operator (`.`) on something that isn't a struct).

| <i>expression</i> | <i>Circle stack, heap or error</i> |
|-------------------|------------------------------------|
| a                 | stack heap error                   |
| *a                | stack heap error                   |
| b                 | stack heap error                   |
| *b                | stack heap error                   |
| c                 | stack heap error                   |
| *c                | stack heap error                   |
| *d                | stack heap error                   |
| *e                | stack heap error                   |
| f                 | stack heap error                   |
| *f                | stack heap error                   |
| g                 | stack heap error                   |
| *g                | stack heap error                   |
| p                 | stack heap error                   |
| *p                | stack heap error                   |
| *q                | stack heap error                   |
| *r                | stack heap error                   |
| *s                | stack heap error                   |
| *t                | stack heap error                   |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int *       |
| *a                |             |
| &a                |             |
| b                 |             |
| *b                |             |
| &b                |             |
| d                 |             |
| *d                |             |
| e                 |             |
| &e                |             |
| p                 |             |
| *r                |             |
| &s                |             |
| t->x              |             |
| q.y               |             |
| q->y              |             |
| (*p).x            |             |
| &(p.x)            |             |
| argc              |             |
| argv[0]           |             |
| argv[0][1]        |             |
| argv              |             |
| &argc             |             |

| <i>expression</i> | <i>Answer</i> |
|-------------------|---------------|
| a                 | stack         |
| *a                | heap          |
| b                 | stack         |
| *b                | stack         |
| c                 | stack         |
| *c                | error         |
| *d                | error         |
| *e                | heap          |
| f                 | stack         |
| *f                | stack         |
| g                 | stack         |
| *g                | error         |
| p                 | stack         |
| *p                | error         |
| *q                | heap          |
| *r                | stack         |
| *s                | stack         |
| *t                | error         |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int *       |
| *a                | int         |
| &a                | int **      |
| b                 | int *       |
| *b                | int         |
| &b                | int**       |
| d                 | double      |
| *d                | error       |
| e                 | double *    |
| &e                | double **   |
| p                 | Point_S     |
| *r                | Point_S     |
| &s                | Point_S **  |
| t->x              | error       |
| q.y               | error       |
| q->y              | int         |
| (*p).x            | error       |
| &(p.x)            | int *       |
| argc              | int         |
| argv[0]           | char *      |
| argv[0][1]        | char        |
| argv              | char **     |
| &argc             | int *       |

End of Key, seed 3456 version **B**  
Total Points: 100

Name \_\_\_\_\_

Please circle your section number:

010 (Wed PM)

011 (Mon PM)

012 (Mon AM)

Answer the multiple choice questions on a “Scantron Form”

Bubble in **ONLY** your Unix userid and your answers

**DO NOT** bubble in your id number or section

If you bubble in your SSN, the computer will **reject your form!!!**

Answer the remaining questions directly on the exam paper.

### General Instructions

- The exam is @@@% multiple choice, and @@@% programming.
- The programming questions start with number 21. You may want to tackle them first, since they may take more time.
- **DO NOT WRITE YOUR NAME ON ANY PAGE EXCEPT THIS ONE!**
- You have 50 minutes. **Pace yourself**, and pay attention to the point values.
- Read *all* the directions *carefully* on each problem.
- Good luck.

Questions 1 through 4 deal with the following code excerpt:

```
1 // foo.cc
2
3 #include <iostream>
4 using namespace std;
5
6 int main(void)
7 {
8 char mascot[10]="Blue Hens";
9 char hometown[20]="Newark";
10 char school[30]="University of Delaware";
11 char state[20]="Delaware";
12
13 char *a = mascot;
14 char * const c = hometown;
15 const char *b = school;
16 const char * const d = state;
17
18
19 cout << a << endl;
20 cout << b << endl;
21 cout << c << endl;
22 cout << d << endl;
23
24 // where some extra code might go
25
26 #include "extra.h"
27
28 cout << a << endl;
29 cout << b << endl;
30 cout << c << endl;
31 cout << d << endl;
32
33 }
34
35
36
37
38
39
40
41
42
43
44
45
46
```

1. (2 pts) One of these lines of code would produce a compiler error if placed at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a[0] = 'F';`
- (b) `b = hometown;`
- (c) `a[1] = 'r';`
- (d) `b[0] = 'D';`
- (e) `a = hometown;`

2. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a = mascot;`
- (b) `a = state;`
- (c) `b[1] = 'u';`
- (d) `c[1] = 'm';`
- (e) `a[1] = 'u';`

3. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `d = state;`
- (b) `b = hometown;`
- (c) `c[0] = 'D';`
- (d) `b = mascot`
- (e) `c[1] = 'i';`

4. (2 pts) Read carefully! Only one of these lines of code would NOT produce a compiler error if placed at the spot indicated by // where some extra code might go ?

That is, which one is CORRECT?

- (a) `c[1] = 'u';`
- (b) `a[1] = 'r';`
- (c) `c = d;`
- (d) `c = a;`
- (e) `d = hometown;`

Questions 5 through 8 refer to the fragment of code shown in Figure 5 that comes from a complete C++ program.

Figure 5: Declaration of a struct

```
...
struct Node
{
 int data;
 Node *next;
};

int main(void)
{
 Node *p;
 Node s;
 p = new Node;
 ...
}
```

5. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 5, at or after the line indicated by . . .  
Which one?
- (a) `data = 3`
  - (b) `s.data = 3;`
  - (c) `(*s).data = 3;`
  - (d) `s->data = 3;`
  - (e) `p.data = 3;`
6. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 5, at or after the line indicated by . . .  
Which one?
- (a) `next = NULL;`
  - (b) `p->next = NULL;`
  - (c) `s->next = NULL;`
  - (d) `p.next = NULL;`
  - (e) `(*s).next = NULL;`

7. (2 pts) Which of the following is equivalent to `p->data`?
- (a) `(*p).data`
  - (b) `*(p.data)`
  - (c) `p.(*data)`
  - (d) `p[data]`
  - (e) `p[1].data`
8. (2 pts) Which of the following is equivalent to `p->next`? (Hint: This is tricky: think carefully about the equivalence of array subscripts and dereferencing of pointers.)
- (a) `*(p.next)`
  - (b) `p[next]`
  - (c) `p[0].next`
  - (d) `p[1].next`
  - (e) `*p[next]`
9. (2 pts) If the variable `a` refers to an array of 10 elements, then the expression `a[0]` is equivalent to which of the following?
- (a) `a`
  - (b) `*a`
  - (c) `a+0`
  - (d) `a++`
  - (e) `++a`
10. (2 pts) Suppose two function definitions in the same program have the same name, but differ in the number and/or type of their formal parameters, as illustrated by these two function prototypes:

```
int max(int a, int b);
int max(int a, int b, int c);
```

This would be an example of which of the following?

- (a) a binary scope resolution operator
- (b) function overloading
- (c) a unary scope resolution operator
- (d) separate compilation
- (e) a syntax error, since this is not permitted

11. (2 pts) If you want the variable `a` to contain the address of an `double`, which of the following would be a correct declaration of `a`?

- (a) `double a;`
- (b) `double &a;`
- (c) `double *a;`
- (d) `double %a;`
- (e) none of the above

12. (2 pts) Suppose you want to write a function `incrModFive` that will increment an integer mod 5 (i.e. if the integer is 4 and you increment it, it will return to 0.) If you want the function `incrModFive` to take one parameter, an `int` passed by reference, and return nothing at all, which of the following would be a correct function prototype for `incrModFive`?

- (a) `incrModFive(int x);`
- (b) `void incrModFive(int x);`
- (c) `void incrModFive(int &x);`
- (d) `int incrModFive(int x);`
- (e) `incrModFive (int &x);`

13. (2 pts) (Continuation of previous question.) If you then called that function, which of the following is a legal call to function `incrModFive`? Assume that the following declarations are in effect:

```
int y,z;
y = 3;
z = 4;
int *x;
x = &y;
```

- (a) `incrModFive(&z);`
- (b) `incrModFive(y);`
- (c) `incrModFive(x);`
- (d) `incrModFive(*y);`
- (e) `z=incrModFive(4);`

14. (2 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?

- (a) `*p`
- (b) `&p`
- (c) `deref(p)`
- (d) `%p`
- (e) `$p`

15. (2 pts) Suppose in a future semester, your instructor for CISC321 requires you to create a web page on strauss with the URL `http://strauss.udel.edu/~userid/CISC321`. Which of the following files could you edit to create the web page?  
(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)
- (a) `~/CISC321/index.html`
  - (b) `~public_html/CISC321/index.html`
  - (c) `~/public_html/cisc321/index.html`
  - (d) `~/public_html/CISC321/index.html`
16. (2 pts) Suppose a UD student named Terry Bly Smart has the userid `tbsmart`. Which of the following URLs brings up his/her personal web page on strauss?  
(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)
- (a) `http://strauss.udel.edu/tbsmart`
  - (b) `http://strauss.udel.edu/~tbsmart`
  - (c) `http://strauss.udel.edu/~tbsmart/public_html`
  - (d) `http://strauss.udel.edu/~tbsmart/public.html`

Here is a sample program, followed by some sample output.—except, the numbers have been replaced with blanks First, trace through the program to determine what goes in each of these blanks, then select the correct answers from the choices given in the problems.

**Don't panic:** Each of the sections of code can be done independently of the other sections, so if you are stuck on one, don't stop; keep working.

```
1 /* ql0.cc Agnes Nitt for CISC181 final exam */
2
3 #include <iostream>
4 using namespace std;
5
6 void trouble(int *r);
7 double fire(double f);
8 int burn(int b, int *n);
9 void cauldron(int &c, int &d);
10
11 int main(void)
12 {
13 int e=2, f=3, t = 4, y = 9;
14 double v=5.1, z = 10.2;
15
16 trouble(&y);
17 cout << "y=" << y << endl;
18
19 z = fire(v);
20 cout << "v=" << v << " z=" << z << endl;
21
22 t = 3;
23 burn(5, &t);
24 cout << "t=" << t << endl;
25
26 e=4;
27 f=5;
28 cauldron(e,f);
29 cout << "e=" << e << " f=" << f << endl;
30 return 0;
31 }
32
33 void trouble(int *r)
34 {
35 (*r) = -1;
36 return;
37 }
38
39 void cauldron(int &c, int &d)
40 {
41 int temp = c;
42 c = d;
43 d = temp;
44 }
45
46 int burn(int b, int *n)
47 {
48 (*n) = b;
49 }
50
51 double fire(double f)
52 {
53 return (f + 0.1);
54 }
```

```
> ./q10
y=___
v=___ z=___
t=___
e=___ f=___
>
```

17. (2 pts) What goes in the blank  $y=$ \_\_\_ ?

- (a)  $y=-1$
- (b)  $y=-9$
- (c)  $y=9$
- (d)  $y=-8$
- (e) none of the above

18. (2 pts) What goes in the blank  $v=$ \_\_\_  $z=$ \_\_\_ ?

- (a)  $v=5.1$   $z=5.1$
- (b)  $v=5.2$   $z=5.2$
- (c)  $v=5.2$   $z=5.1$
- (d)  $v=5.1$   $z=5.2$
- (e) none of the above

19. (2 pts) What goes in the blank  $t=$ \_\_\_ ?

- (a)  $t=3$
- (b)  $t=5$
- (c)  $t=8$
- (d)  $t=15$
- (e) none of the above

20. (2 pts) What goes in the blank  $e=$ \_\_\_  $f=$ \_\_\_ ?

- (a)  $e=5$   $f=4$
- (b)  $e=4$   $f=5$
- (c)  $e=5$   $f=5$
- (d)  $e=4$   $f=4$
- (e) none of the above

21. (20 pts) The main program listed below demonstrates a function that can put an array of temperature values in order from hottest to coldest. Sample output is given, along with a complete main program, and three function definitions. The last two function definitions on the following page are incomplete. Finish them, using the hints provided. Be extra careful with the order of parameters.

Sample output:

```
> ./a.out
Temps: 45.0 32.0 54.0 8.5 19.1
Hottest to Coldest: 54.0 45.0 32.0 19.1 8.5
>
```

Code:

```
/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */
/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}
```

```

void outputArrayOnOneLine(int count, double array[])
{
 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?

 About 3-4 lines of code is all you need.
 */

}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array

 About 6-8 lines of code will be enough. */

 int j;
 double temp;
 int indexOfColdest = 0; /* index of coldest element */

}

```

The following questions deal with this code excerpt:

```
// quiz question
#include <iostream>
using namespace std;

struct Point_S
{
 int x;
 int y;
};

int main(int argc, char*argv[])

{
 int *a;
 int b;
 int *c;

 double *d;
 double *e;
 double f;
 double g;

 Point_S p;
 Point_S q;
 Point_S *r;
 Point_S *s;
 Point_S *t;

 c = new int;
 a=&b;

 d = new double;
 e=&f;

 r = new Point_S;
 s = new Point_S;
 t=&p;

 cout << "Hi" << endl;
 return 0;
}
```

22. (40 pts) Suppose you run the C++ program on the previous page, and the program reaches the line of code that prints out Hi.

Indicate whether, at that point in time, the expression given in the table on the left

- refers to a memory location on the *stack*,
- refers to a memory location on the *heap*,
- would cause an *error* (e.g. dereferencing something that isn't a pointer.)

Then, in the table on the right, fill in the type of each expression, or write "error" if the expression would not be valid (e.g. dereferencing something that isn't a pointer, or using the dot operator (.) on something that isn't a struct.

| <i>expression</i> | <i>Circle stack, heap or error</i> |
|-------------------|------------------------------------|
| a                 | stack heap error                   |
| *a                | stack heap error                   |
| b                 | stack heap error                   |
| *b                | stack heap error                   |
| c                 | stack heap error                   |
| *c                | stack heap error                   |
| *d                | stack heap error                   |
| *e                | stack heap error                   |
| f                 | stack heap error                   |
| *f                | stack heap error                   |
| g                 | stack heap error                   |
| *g                | stack heap error                   |
| p                 | stack heap error                   |
| *p                | stack heap error                   |
| *q                | stack heap error                   |
| *r                | stack heap error                   |
| *s                | stack heap error                   |
| *t                | stack heap error                   |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int *       |
| *a                |             |
| &a                |             |
| b                 |             |
| *b                |             |
| &b                |             |
| d                 |             |
| *d                |             |
| e                 |             |
| &e                |             |
| p                 |             |
| *r                |             |
| &s                |             |
| t->x              |             |
| q.y               |             |
| q->y              |             |
| (*p).x            |             |
| &(p.x)            |             |
| argc              |             |
| argv[0]           |             |
| argv[0][1]        |             |
| argv              |             |
| &argc             |             |

End of Exam. Total Points: 100

XYAFRC93DAPE38B82D6BF3N

**C** KEY CISC 181 sections 010-012, Midterm 2

11/11/05

Questions 1 through 4 deal with the following code excerpt:

```
1 // foo.cc
2
3 #include <iostream>
4 using namespace std;
5
6 int main(void)
7 {
8 char mascot[10]="Blue Hens";
9 char hometown[20]="Newark";
10 char school[30]="University of Delaware";
11 char state[20]="Delaware";
12
13 char *a = mascot;
14 char * const c = hometown;
15 const char *b = school;
16 const char * const d = state;
17
18
19 cout << a << endl;
20 cout << b << endl;
21 cout << c << endl;
22 cout << d << endl;
23
24 // where some extra code might go
25
26 #include "extra.h"
27
28 cout << a << endl;
29 cout << b << endl;
30 cout << c << endl;
31 cout << d << endl;
32
33 }
34
35
36
37
38
39
40
41
42
43
44
45
46
```

1. (2 pts) One of these lines of code would produce a compiler error if placed at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a[0] = 'F';`
- (b) `b = hometown;`
- (c) `a[1] = 'r';`
- (d) `b[0] = 'D';`
- (e) `a = hometown;`

2. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a = mascot;`
- (b) `a = state;`
- (c) `b[1] = 'u';`
- (d) `c[1] = 'm';`
- (e) `a[1] = 'u';`

3. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `d = state;`
- (b) `b = hometown;`
- (c) `c[0] = 'D';`
- (d) `b = mascot`
- (e) `c[1] = 'i';`

4. (2 pts) Read carefully! Only one of these lines of code would NOT produce a compiler error if placed at the spot indicated by // where some extra code might go ?

That is, which one is CORRECT?

- (a) `c[1] = 'u';`
- (b) `a[1] = 'r';`
- (c) `c = d;`
- (d) `c = a;`
- (e) `d = hometown;`

Questions 5 through 8 refer to the fragment of code shown in Figure 5 that comes from a complete C++ program.

Figure 6: Declaration of a struct

```
...
struct Node
{
 int data;
 Node *next;
};

int main(void)
{
 Node *p;
 Node s;
 p = new Node;
 ...
}
```

5. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 5, at or after the line indicated by ...  
Which one?
- (a) data = 3
  - (b) s.data = 3;
  - (c) (\*s).data = 3;
  - (d) s->data = 3;
  - (e) p.data = 3;
6. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 5, at or after the line indicated by ...  
Which one?
- (a) next = NULL;
  - (b) p->next = NULL;
  - (c) s->next = NULL;
  - (d) p.next = NULL;
  - (e) (\*s).next = NULL;
7. (2 pts) Which of the following is equivalent to p->data?
- (a) (\*p).data
  - (b) \*(p.data)
  - (c) p.(\*data)
  - (d) p[data]
  - (e) p[1].data

8. (2 pts) Which of the following is equivalent to `p->next`? (Hint: This is tricky: think carefully about the equivalence of array subscripts and dereferencing of pointers.)

- (a) `*(p.next)`
- (b) `p[next]`
- (c) `p[0].next`
- (d) `p[1].next`
- (e) `*p[next]`

9. (2 pts) If the variable `a` refers to an array of 10 elements, then the expression `a[0]` is equivalent to which of the following?

- (a) `a`
- (b) `*a`
- (c) `a+0`
- (d) `a++`
- (e) `++a`

10. (2 pts) Suppose two function definitions in the same program have the same name, but differ in the number and/or type of their formal parameters, as illustrated by these two function prototypes:

```
int max(int a, int b);
int max(int a, int b, int c);
```

This would be an example of which of the following?

- (a) a binary scope resolution operator
- (b) function overloading
- (c) a unary scope resolution operator
- (d) separate compilation
- (e) a syntax error, since this is not permitted

11. (2 pts) If you want the variable `a` to contain the address of an `double`, which of the following would be a correct declaration of `a`?

- (a) `double a;`
- (b) `double &a;`
- (c) `double *a;`
- (d) `double %a;`
- (e) none of the above

12. (2 pts) Suppose you want to write a function `incrModFive` that will increment an integer mod 5 (i.e. if the integer is 4 and you increment it, it will return to 0.) If you want the function `incrModFive` to take one parameter, an `int` passed by reference, and return nothing at all, which of the following would be a correct function prototype for `incrModFive`?

- (a) `incrModFive(int x);`
- (b) `void incrModFive(int x);`
- (c) `void incrModFive(int &x);`
- (d) `int incrModFive(int x);`
- (e) `incrModFive (int &x);`

13. (2 pts) (Continuation of previous question.) If you then called that function, which of the following is a legal call to function `incrModFive`? Assume that the following declarations are in effect:

```
int y, z;
y = 3;
z = 4;
int *x;
x = &y;
```

- (a) `incrModFive(&z);`
- (b) `incrModFive(y);`
- (c) `incrModFive(x);`
- (d) `incrModFive(*y);`
- (e) `z=incrModFive(4);`

14. (2 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?

- (a) `*p`
- (b) `&p`
- (c) `deref(p)`
- (d) `%p`
- (e) `$p`

15. (2 pts) Suppose in a future semester, your instructor for CISC321 requires you to create a web page on `strauss` with the URL

`http://strauss.udel.edu/~userid/CISC321`. Which of the following files could you edit to create the web page?

(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)

- (a) `~/CISC321/index.html`
- (b) `~/public_html/CISC321/index.html`
- (c) `~/public_html/cisc321/index.html`
- (d) `~/public_html/CISC321/index.html`

16. (2 pts) Suppose a UD student named Terry Bly Smart has the userid tbsmart. Which of the following URLs brings up his/her personal web page on strauss?

(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)

- (a) `http://strauss.udel.edu/tbsmart`
- (b) `http://strauss.udel.edu/~tbsmart`
- (c) `http://strauss.udel.edu/~tbsmart/public.html`
- (d) `http://strauss.udel.edu/~tbsmart/public.html`

Here is a sample program, followed by some sample output.—except, the numbers have been replaced with blanks First, trace through the program to determine what goes in each of these blanks, then select the correct answers from the choices given in the problems.

**Don't panic:** Each of the sections of code can be done independently of the other sections, so if you are stuck on one, don't stop; keep working.

```
1 /* q10.cc Agnes Nitt for CISC181 final exam */
2
3 #include <iostream>
4 using namespace std;
5
6 void trouble(int *r);
7 double fire(double f);
8 int burn(int b, int *n);
9 void cauldron(int &c, int &d);
10
11 int main(void)
12 {
13 int e=2, f=3, t = 4, y = 9;
14 double v=5.1, z = 10.2;
15
16 trouble(&y);
17 cout << "y=" << y << endl;
18
19 z = fire(v);
20 cout << "v=" << v << " z=" << z << endl;
21
22 t = 3;
23 burn(5, &t);
24 cout << "t=" << t << endl;
25
26 e=4;
27 f=5;
28 cauldron(e,f);
29 cout << "e=" << e << " f=" << f << endl;
30 return 0;
31 }
32
33 void trouble(int *r)
34 {
35 (*r) = -1;
36 return;
37 }
38
39 void cauldron(int &c, int &d)
40 {
41 int temp = c;
42 c = d;
43 d = temp;
44 }
```

```
45 |
46 | int burn(int b, int *n)
47 | {
48 | (*n) = b;
49 | }
50 |
51 | double fire(double f)
52 | {
53 | return (f + 0.1);
54 | }
```

```
> ./q10
y=___
v=___ z=___
t=___
e=___ f=___
>
```

17. (2 pts) What goes in the blank  $y=$ \_\_\_ ?

- (a)  $y=-1$
- (b)  $y=-9$
- (c)  $y=9$
- (d)  $y=-8$
- (e) none of the above

18. (2 pts) What goes in the blank  $v=$ \_\_\_  $z=$ \_\_\_ ?

- (a)  $v=5.1$   $z=5.1$
- (b)  $v=5.2$   $z=5.2$
- (c)  $v=5.2$   $z=5.1$
- (d)  $v=5.1$   $z=5.2$
- (e) none of the above

19. (2 pts) What goes in the blank  $t=$ \_\_\_ ?

- (a)  $t=3$
- (b)  $t=5$
- (c)  $t=8$
- (d)  $t=15$
- (e) none of the above

20. (2 pts) What goes in the blank  $e=$ \_\_\_  $f=$ \_\_\_ ?

- (a)  $e=5$   $f=4$
- (b)  $e=4$   $f=5$
- (c)  $e=5$   $f=5$
- (d)  $e=4$   $f=4$
- (e) none of the above

21. (20 pts) The main program listed below demonstrates a function that can put an array of temperature values in order from hottest to coldest. Sample output is given, along with a complete main program, and three function definitions. The last two function definitions on the following page are incomplete. Finish them, using the hints provided. Be extra careful with the order of parameters.

Sample output:

```
> ./a.out
Temps: 45.0 32.0 54.0 8.5 19.1
Hottest to Coldest: 54.0 45.0 32.0 19.1 8.5
>
```

Code:

```
/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */
/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}

void outputArrayOnOneLine(int count, double array[])
{
```

```

 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?

 About 3-4 lines of code is all you need.
 */
}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array

 About 6-8 lines of code will be enough. */

 int j;
 double temp;
 int indexOfColdest = 0; /* index of coldest element */

}

/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */

```

```

/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}

void outputArrayOnOneLine(int count, double array[])
{
 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?
 */

 for (int i=howMany; i>1; i--)
 {
 putColdestAtEnd(temps, i);
 }
}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array */

```

```

int j;
double temp;
int indexOfColdest = 0; /* index of coldest element */

for (j=1; j<numTemps; j++)
{
 if (tempArray[j] < tempArray[indexOfColdest])
 {
 indexOfColdest = j;
 }
}

temp = tempArray[indexOfColdest];
tempArray[indexOfColdest] = tempArray[numTemps - 1];
tempArray[numTemps - 1] = temp;
}

```

The following questions deal with this code excerpt:

```

// quiz question
#include <iostream>
using namespace std;

struct Point_S
{
 int x;
 int y;
};

int main(int argc, char*argv[])

{
 int *a;
 int b;
 int *c;

 double *d;
 double *e;
 double f;
 double g;

 Point_S p;
 Point_S q;
 Point_S *r;
 Point_S *s;
 Point_S *t;
}

```

```

c = new int;
a=&b;

d = new double;
e=&f;

r = new Point_S;
s = new Point_S;
t=&p;

cout << "Hi" << endl;
return 0;
}

```

22. (40 pts) Suppose you run the C++ program on the previous page, and the program reaches the line of code that prints out `Hi`.

Indicate whether, at that point in time, the expression given in the table on the left

- refers to a memory location on the *stack*,
- refers to a memory location on the *heap*,
- would cause an *error* (e.g. dereferencing something that isn't a pointer.)

Then, in the table on the right, fill in the type of each expression, or write "error" if the expression would not be valid (e.g. dereferencing something that isn't a pointer, or using the dot operator (`.`) on something that isn't a struct).

| <i>expression</i> | <i>Circle stack, heap or error</i> |
|-------------------|------------------------------------|
| a                 | stack heap error                   |
| *a                | stack heap error                   |
| b                 | stack heap error                   |
| *b                | stack heap error                   |
| c                 | stack heap error                   |
| *c                | stack heap error                   |
| *d                | stack heap error                   |
| *e                | stack heap error                   |
| f                 | stack heap error                   |
| *f                | stack heap error                   |
| g                 | stack heap error                   |
| *g                | stack heap error                   |
| p                 | stack heap error                   |
| *p                | stack heap error                   |
| *q                | stack heap error                   |
| *r                | stack heap error                   |
| *s                | stack heap error                   |
| *t                | stack heap error                   |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int *       |
| *a                |             |
| &a                |             |
| b                 |             |
| *b                |             |
| &b                |             |
| d                 |             |
| *d                |             |
| e                 |             |
| &e                |             |
| p                 |             |
| *r                |             |
| &s                |             |
| t->x              |             |
| q.y               |             |
| q->y              |             |
| (*p).x            |             |
| &(p.x)            |             |
| argc              |             |
| argv[0]           |             |
| argv[0][1]        |             |
| argv              |             |
| &argc             |             |

| <i>expression</i> | <i>Answer</i> |
|-------------------|---------------|
| a                 | stack         |
| *a                | stack         |
| b                 | stack         |
| *b                | error         |
| c                 | stack         |
| *c                | heap          |
| *d                | heap          |
| *e                | stack         |
| f                 | stack         |
| *f                | error         |
| g                 | stack         |
| *g                | error         |
| p                 | stack         |
| *p                | error         |
| *q                | error         |
| *r                | heap          |
| *s                | heap          |
| *t                | stack         |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int *       |
| *a                | int         |
| &a                | int **      |
| b                 | int         |
| *b                | error       |
| &b                | int *       |
| d                 | double *    |
| *d                | double      |
| e                 | double *    |
| &e                | double **   |
| p                 | Point_S     |
| *r                | Point_S     |
| &s                | Point_S **  |
| t->x              | int         |
| q.y               | int         |
| q->y              | error       |
| (*p).x            | error       |
| &(p.x)            | int *       |
| argc              | int         |
| argv[0]           | char *      |
| argv[0][1]        | char        |
| argv              | char **     |
| &argc             | int *       |

End of Key, seed 3456 version **C**  
Total Points: 100

Name \_\_\_\_\_

Please circle your section number:

010 (Wed PM)

011 (Mon PM)

012 (Mon AM)

Answer the multiple choice questions on a “Scantron Form”

Bubble in **ONLY** your Unix userid and your answers

**DO NOT** bubble in your id number or section

If you bubble in your SSN, the computer will **reject your form!!!**

Answer the remaining questions directly on the exam paper.

### General Instructions

- The exam is @@@% multiple choice, and @@@% programming.
- The programming questions start with number 21. You may want to tackle them first, since they may take more time.
- **DO NOT WRITE YOUR NAME ON ANY PAGE EXCEPT THIS ONE!**
- You have 50 minutes. **Pace yourself**, and pay attention to the point values.
- Read *all* the directions *carefully* on each problem.
- Good luck.

Questions 1 through 4 deal with the following code excerpt:

```
1 // foo.cc
2
3 #include <iostream>
4 using namespace std;
5
6 int main(void)
7 {
8 char mascot[10]="Blue Hens";
9 char hometown[20]="Newark";
10 char school[30]="University of Delaware";
11 char state[20]="Delaware";
12
13 char *a = mascot;
14 char * const c = hometown;
15 const char *b = school;
16 const char * const d = state;
17
18
19 cout << a << endl;
20 cout << b << endl;
21 cout << c << endl;
22 cout << d << endl;
23
24 // where some extra code might go
25
26 #include "extra.h"
27
28 cout << a << endl;
29 cout << b << endl;
30 cout << c << endl;
31 cout << d << endl;
32
33 }
34
35
36
37
38
39
40
41
42
43
44
45
46
```

1. (2 pts) One of these lines of code would produce a compiler error if placed at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a[0] = 'F';`
- (b) `b = hometown;`
- (c) `a[1] = 'r';`
- (d) `b[0] = 'D';`
- (e) `a = hometown;`

2. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a = mascot;`
- (b) `a = state;`
- (c) `b[1] = 'u';`
- (d) `c[1] = 'm';`
- (e) `a[1] = 'u';`

3. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `d = state;`
- (b) `b = hometown;`
- (c) `c[0] = 'D';`
- (d) `b = mascot`
- (e) `c[1] = 'i';`

4. (2 pts) Read carefully! Only one of these lines of code would NOT produce a compiler error if placed at the spot indicated by // where some extra code might go ?

That is, which one is CORRECT?

- (a) `c[1] = 'u';`
- (b) `a[1] = 'r';`
- (c) `c = d;`
- (d) `c = a;`
- (e) `d = hometown;`

Questions 5 through 8 refer to the fragment of code shown in Figure 7 that comes from a complete C++ program.

Figure 7: Declaration of a struct

```
...
struct Node
{
 int data;
 Node *next;
};

int main(void)
{
 Node *p;
 Node s;
 p = new Node;
 ...
}
```

5. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 7, at or after the line indicated by . . .  
Which one?
- (a) `data = 3`
  - (b) `s.data = 3;`
  - (c) `(*s).data = 3;`
  - (d) `s->data = 3;`
  - (e) `p.data = 3;`
6. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 7, at or after the line indicated by . . .  
Which one?
- (a) `next = NULL;`
  - (b) `p->next = NULL;`
  - (c) `s->next = NULL;`
  - (d) `p.next = NULL;`
  - (e) `(*s).next = NULL;`

7. (2 pts) Which of the following is equivalent to `p->data`?
- (a) `(*p).data`
  - (b) `*(p.data)`
  - (c) `p.(*data)`
  - (d) `p[data]`
  - (e) `p[1].data`
8. (2 pts) Which of the following is equivalent to `p->next`? (Hint: This is tricky: think carefully about the equivalence of array subscripts and dereferencing of pointers.)
- (a) `*(p.next)`
  - (b) `p[next]`
  - (c) `p[0].next`
  - (d) `p[1].next`
  - (e) `*p[next]`
9. (2 pts) If the variable `a` refers to an array of 10 elements, then the expression `a[0]` is equivalent to which of the following?
- (a) `a`
  - (b) `*a`
  - (c) `a+0`
  - (d) `a++`
  - (e) `++a`
10. (2 pts) Suppose two function definitions in the same program have the same name, but differ in the number and/or type of their formal parameters, as illustrated by these two function prototypes:

```
int max(int a, int b);
int max(int a, int b, int c);
```

This would be an example of which of the following?

- (a) a binary scope resolution operator
- (b) function overloading
- (c) a unary scope resolution operator
- (d) separate compilation
- (e) a syntax error, since this is not permitted

11. (2 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?
- (a) `*p`
  - (b) `&p`
  - (c) `deref(p)`
  - (d) `%p`
  - (e) `$p`
12. (2 pts) If you want the variable `a` to contain the address of an `double`, which of the following would be a correct declaration of `a`?
- (a) `double a;`
  - (b) `double &a;`
  - (c) `double *a;`
  - (d) `double %a;`
  - (e) none of the above
13. (2 pts) Suppose you want to write a function `incrModFive` that will increment an integer mod 5 (i.e. if the integer is 4 and you increment it, it will return to 0.) If you want the function `incrModFive` to take one parameter, an `int` passed by reference, and return nothing at all, which of the following would be a correct function prototype for `incrModFive`?
- (a) `incrModFive(int x);`
  - (b) `void incrModFive(int x);`
  - (c) `void incrModFive(int &x);`
  - (d) `int incrModFive(int x);`
  - (e) `incrModFive (int &x);`
14. (2 pts) (Continuation of previous question.) If you then called that function, which of the following is a legal call to function `incrModFive`? Assume that the following declarations are in effect:

```
int y,z;
y = 3;
z = 4;
int *x;
x = &y;
```

- (a) `incrModFive(&z);`
- (b) `incrModFive(y);`
- (c) `incrModFive(x);`
- (d) `incrModFive(*y);`
- (e) `z=incrModFive(4);`

15. (2 pts) Suppose in a future semester, your instructor for CISC321 requires you to create a web page on strauss with the URL `http://strauss.udel.edu/~userid/CISC321`. Which of the following files could you edit to create the web page?  
(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)
- (a) `~/CISC321/index.html`
  - (b) `~public_html/CISC321/index.html`
  - (c) `~/public_html/cisc321/index.html`
  - (d) `~/public_html/CISC321/index.html`
16. (2 pts) Suppose a UD student named Terry Bly Smart has the userid `tbsmart`. Which of the following URLs brings up his/her personal web page on strauss?  
(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)
- (a) `http://strauss.udel.edu/tbsmart`
  - (b) `http://strauss.udel.edu/~tbsmart`
  - (c) `http://strauss.udel.edu/~tbsmart/public_html`
  - (d) `http://strauss.udel.edu/~tbsmart/public.html`

Here is a sample program, followed by some sample output.—except, the numbers have been replaced with blanks First, trace through the program to determine what goes in each of these blanks, then select the correct answers from the choices given in the problems.

**Don't panic:** Each of the sections of code can be done independently of the other sections, so if you are stuck on one, don't stop; keep working.

```
1 /* ql0.cc Agnes Nitt for CISC181 final exam */
2
3 #include <iostream>
4 using namespace std;
5
6 void trouble(int *r);
7 double fire(double f);
8 int burn(int b, int *n);
9 void cauldron(int &c, int &d);
10
11 int main(void)
12 {
13 int e=2, f=3, t = 4, y = 9;
14 double v=5.1, z = 10.2;
15
16 trouble(&y);
17 cout << "y=" << y << endl;
18
19 z = fire(v);
20 cout << "v=" << v << " z=" << z << endl;
21
22 t = 3;
23 burn(5, &t);
24 cout << "t=" << t << endl;
25
26 e=4;
27 f=5;
28 cauldron(e,f);
29 cout << "e=" << e << " f=" << f << endl;
30 return 0;
31 }
32
33 void trouble(int *r)
34 {
35 (*r) = -1;
36 return;
37 }
38
39 void cauldron(int &c, int &d)
40 {
41 int temp = c;
42 c = d;
43 d = temp;
44 }
45
46 int burn(int b, int *n)
47 {
48 (*n) = b;
49 }
50
51 double fire(double f)
52 {
53 return (f + 0.1);
54 }
```

```
> ./q10
y=___
v=___ z=___
t=___
e=___ f=___
>
```

17. (2 pts) What goes in the blank  $y=$ \_\_\_ ?

- (a)  $y=-1$
- (b)  $y=-9$
- (c)  $y=9$
- (d)  $y=-8$
- (e) none of the above

18. (2 pts) What goes in the blank  $v=$ \_\_\_  $z=$ \_\_\_ ?

- (a)  $v=5.1$   $z=5.1$
- (b)  $v=5.2$   $z=5.2$
- (c)  $v=5.2$   $z=5.1$
- (d)  $v=5.1$   $z=5.2$
- (e) none of the above

19. (2 pts) What goes in the blank  $t=$ \_\_\_ ?

- (a)  $t=3$
- (b)  $t=5$
- (c)  $t=8$
- (d)  $t=15$
- (e) none of the above

20. (2 pts) What goes in the blank  $e=$ \_\_\_  $f=$ \_\_\_ ?

- (a)  $e=5$   $f=4$
- (b)  $e=4$   $f=5$
- (c)  $e=5$   $f=5$
- (d)  $e=4$   $f=4$
- (e) none of the above

21. (20 pts) The main program listed below demonstrates a function that can put an array of temperature values in order from hottest to coldest. Sample output is given, along with a complete main program, and three function definitions. The last two function definitions on the following page are incomplete. Finish them, using the hints provided. Be extra careful with the order of parameters.

Sample output:

```
> ./a.out
Temps: 45.0 32.0 54.0 8.5 19.1
Hottest to Coldest: 54.0 45.0 32.0 19.1 8.5
>
```

Code:

```
/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */
/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}
```

```

void outputArrayOnOneLine(int count, double array[])
{
 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?

 About 3-4 lines of code is all you need.
 */

}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array

 About 6-8 lines of code will be enough. */

 int j;
 double temp;
 int indexOfColdest = 0; /* index of coldest element */

}

```

The following questions deal with this code excerpt:

```
// quiz question
#include <iostream>
using namespace std;

struct Point_S
{
 int x;
 int y;
};

int main(int argc, char*argv[])

{
 int a;
 int *b;
 int *c;

 double *d;
 double e;
 double f;
 double *g;

 Point_S *p;
 Point_S q;
 Point_S r;
 Point_S *s;
 Point_S *t;

 b = new int;
 c = &a;

 g=new double
 d=&e;

 s = new Point_S;
 t = &r;
 p=&q;

 cout << "Hi" << endl;
 return 0;
}
```

22. (40 pts) Suppose you run the C++ program on the previous page, and the program reaches the line of code that prints out Hi.

Indicate whether, at that point in time, the expression given in the table on the left

- refers to a memory location on the *stack*,
- refers to a memory location on the *heap*,
- would cause an *error* (e.g. dereferencing something that isn't a pointer.)

Then, in the table on the right, fill in the type of each expression, or write "error" if the expression would not be valid (e.g. dereferencing something that isn't a pointer, or using the dot operator (.) on something that isn't a struct.

| <i>expression</i> | <i>Circle stack, heap or error</i> |
|-------------------|------------------------------------|
| a                 | stack heap error                   |
| *a                | stack heap error                   |
| b                 | stack heap error                   |
| *b                | stack heap error                   |
| c                 | stack heap error                   |
| *c                | stack heap error                   |
| *d                | stack heap error                   |
| *e                | stack heap error                   |
| f                 | stack heap error                   |
| *f                | stack heap error                   |
| g                 | stack heap error                   |
| *g                | stack heap error                   |
| p                 | stack heap error                   |
| *p                | stack heap error                   |
| *q                | stack heap error                   |
| *r                | stack heap error                   |
| *s                | stack heap error                   |
| *t                | stack heap error                   |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int         |
| *a                |             |
| &a                |             |
| b                 |             |
| *b                |             |
| &b                |             |
| d                 |             |
| *d                |             |
| e                 |             |
| &e                |             |
| p                 |             |
| *r                |             |
| &s                |             |
| t->x              |             |
| q.y               |             |
| q->y              |             |
| (*p).x            |             |
| &(p.x)            |             |
| argc              |             |
| argv[0]           |             |
| argv[0][1]        |             |
| argv              |             |
| &argc             |             |

End of Exam. Total Points: 100

XYBFRD94BBPE38A82D6AF3S

**D** KEY CISC 181 sections 010-012, Midterm 2

11/11/05

Questions 1 through 4 deal with the following code excerpt:

```
1 // foo.cc
2
3 #include <iostream>
4 using namespace std;
5
6 int main(void)
7 {
8 char mascot[10]="Blue Hens";
9 char hometown[20]="Newark";
10 char school[30]="University of Delaware";
11 char state[20]="Delaware";
12
13 char *a = mascot;
14 char * const c = hometown;
15 const char *b = school;
16 const char * const d = state;
17
18
19 cout << a << endl;
20 cout << b << endl;
21 cout << c << endl;
22 cout << d << endl;
23
24 // where some extra code might go
25
26 #include "extra.h"
27
28 cout << a << endl;
29 cout << b << endl;
30 cout << c << endl;
31 cout << d << endl;
32
33 }
34
35
36
37
38
39
40
41
42
43
44
45
46
```

1. (2 pts) One of these lines of code would produce a compiler error if placed at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a[0] = 'F';`
- (b) `b = hometown;`
- (c) `a[1] = 'r';`
- (d) `b[0] = 'D';`
- (e) `a = hometown;`

2. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `a = mascot;`
- (b) `a = state;`
- (c) `b[1] = 'u';`
- (d) `c[1] = 'm';`
- (e) `a[1] = 'u';`

3. (2 pts) One of these lines of code would produce a compiler error if place at the spot indicated by // where some extra code might go ?

Which one is WRONG?

- (a) `d = state;`
- (b) `b = hometown;`
- (c) `c[0] = 'D';`
- (d) `b = mascot`
- (e) `c[1] = 'i';`

4. (2 pts) Read carefully! Only one of these lines of code would NOT produce a compiler error if placed at the spot indicated by // where some extra code might go ?

That is, which one is CORRECT?

- (a) `c[1] = 'u';`
- (b) `a[1] = 'r';`
- (c) `c = d;`
- (d) `c = a;`
- (e) `d = hometown;`

Questions 5 through 8 refer to the fragment of code shown in Figure 7 that comes from a complete C++ program.

Figure 8: Declaration of a struct

```
...
struct Node
{
 int data;
 Node *next;
};

int main(void)
{
 Node *p;
 Node s;
 p = new Node;
 ...
}
```

5. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 7, at or after the line indicated by ...  
Which one?
- (a) data = 3
  - (b) s.data = 3;
  - (c) (\*s).data = 3;
  - (d) s->data = 3;
  - (e) p.data = 3;
6. (2 pts) Only one of the following lines could legally appear in the program referred to in Figure 7, at or after the line indicated by ...  
Which one?
- (a) next = NULL;
  - (b) p->next = NULL;
  - (c) s->next = NULL;
  - (d) p.next = NULL;
  - (e) (\*s).next = NULL;
7. (2 pts) Which of the following is equivalent to p->data?
- (a) (\*p).data
  - (b) \*(p.data)
  - (c) p.(\*data)
  - (d) p[data]
  - (e) p[1].data

8. (2 pts) Which of the following is equivalent to `p->next`? (Hint: This is tricky: think carefully about the equivalence of array subscripts and dereferencing of pointers.)

- (a) `*(p.next)`
- (b) `p[next]`
- (c) `p[0].next`
- (d) `p[1].next`
- (e) `*p[next]`

9. (2 pts) If the variable `a` refers to an array of 10 elements, then the expression `a[0]` is equivalent to which of the following?

- (a) `a`
- (b) `*a`
- (c) `a+0`
- (d) `a++`
- (e) `++a`

10. (2 pts) Suppose two function definitions in the same program have the same name, but differ in the number and/or type of their formal parameters, as illustrated by these two function prototypes:

```
int max(int a, int b);
int max(int a, int b, int c);
```

This would be an example of which of the following?

- (a) a binary scope resolution operator
- (b) function overloading
- (c) a unary scope resolution operator
- (d) separate compilation
- (e) a syntax error, since this is not permitted

11. (2 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?

- (a) `*p`
- (b) `&p`
- (c) `deref(p)`
- (d) `%p`
- (e) `$p`

12. (2 pts) If you want the variable `a` to contain the address of an `double`, which of the following would be a correct declaration of `a`?

- (a) `double a;`
- (b) `double &a;`
- (c) `double *a;`
- (d) `double %a;`
- (e) none of the above

13. (2 pts) Suppose you want to write a function `incrModFive` that will increment an integer mod 5 (i.e. if the integer is 4 and you increment it, it will return to 0.) If you want the function `incrModFive` to take one parameter, an `int` passed by reference, and return nothing at all, which of the following would be a correct function prototype for `incrModFive`?

(a) `incrModFive(int x);`  
(b) `void incrModFive(int x);`  
 (c) `void incrModFive(int &x);`  
(d) `int incrModFive(int x);`  
(e) `incrModFive (int &x);`

14. (2 pts) (Continuation of previous question.) If you then called that function, which of the following is a legal call to function `incrModFive`? Assume that the following declarations are in effect:

```
int y, z;
y = 3;
z = 4;
int *x;
x = &y;
```

(a) `incrModFive(&z);`  
 (b) `incrModFive(y);`  
(c) `incrModFive(x);`  
(d) `incrModFive(*y);`  
(e) `z=incrModFive(4);`

15. (2 pts) Suppose in a future semester, your instructor for CISC321 requires you to create a web page on `strauss` with the URL `http://strauss.udel.edu/~userid/CISC321`. Which of the following files could you edit to create the web page?

(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)

(a) `~/CISC321/index.html`  
(b) `~/public_html/CISC321/index.html`  
(c) `~/public_html/cisc321/index.html`  
 (d) `~/public_html/CISC321/index.html`

16. (2 pts) Suppose a UD student named Terry Bly Smart has the userid `tbsmart`. Which of the following URLs brings up his/her personal web page on `strauss`?

(Note: The prefixes `http://udel.edu`, `http://copland.udel.edu` and `strauss.udel.edu` all point to the same set of web pages, so don't worry about that part of the answer.)

(a) `http://strauss.udel.edu/tbsmart`  
 (b) `http://strauss.udel.edu/~tbsmart`  
(c) `http://strauss.udel.edu/~tbsmart/public.html`  
(d) `http://strauss.udel.edu/~tbsmart/public.html`

Here is a sample program, followed by some sample output.—except, the numbers have been replaced with blanks First, trace through the program to determine what goes in each of these blanks, then select the correct answers from the choices given in the problems.

**Don't panic:** Each of the sections of code can be done independently of the other sections, so if you are stuck on one, don't stop; keep working.

```
1 /* q10.cc Agnes Nitt for CISC181 final exam */
2
3 #include <iostream>
4 using namespace std;
5
6 void trouble(int *r);
7 double fire(double f);
8 int burn(int b, int *n);
9 void cauldron(int &c, int &d);
10
11 int main(void)
12 {
13 int e=2, f=3, t = 4, y = 9;
14 double v=5.1, z = 10.2;
15
16 trouble(&y);
17 cout << "y=" << y << endl;
18
19 z = fire(v);
20 cout << "v=" << v << " z=" << z << endl;
21
22 t = 3;
23 burn(5, &t);
24 cout << "t=" << t << endl;
25
26 e=4;
27 f=5;
28 cauldron(e,f);
29 cout << "e=" << e << " f=" << f << endl;
30 return 0;
31 }
32
33 void trouble(int *r)
34 {
35 (*r) = -1;
36 return;
37 }
38
39 void cauldron(int &c, int &d)
40 {
41 int temp = c;
42 c = d;
43 d = temp;
44 }
45
46 int burn(int b, int *n)
47 {
48 (*n) = b;
49 }
50
51 double fire(double f)
52 {
53 return (f + 0.1);
54 }
```

> ./q10

y=\_\_\_\_

v=\_\_\_\_ z=\_\_\_\_

t=\_\_\_\_  
e=\_\_\_\_ f=\_\_\_\_  
>

17. (2 pts) What goes in the blank  $y=$ \_\_\_ ?

- (a)  $y=-1$
- (b)  $y=-9$
- (c)  $y=9$
- (d)  $y=-8$
- (e) none of the above

18. (2 pts) What goes in the blank  $v=$ \_\_\_  $z=$ \_\_\_ ?

- (a)  $v=5.1$   $z=5.1$
- (b)  $v=5.2$   $z=5.2$
- (c)  $v=5.2$   $z=5.1$
- (d)  $v=5.1$   $z=5.2$
- (e) none of the above

19. (2 pts) What goes in the blank  $t=$ \_\_\_ ?

- (a)  $t=3$
- (b)  $t=5$
- (c)  $t=8$
- (d)  $t=15$
- (e) none of the above

20. (2 pts) What goes in the blank  $e=$ \_\_\_  $f=$ \_\_\_ ?

- (a)  $e=5$   $f=4$
- (b)  $e=4$   $f=5$
- (c)  $e=5$   $f=5$
- (d)  $e=4$   $f=4$
- (e) none of the above

21. (20 pts) The main program listed below demonstrates a function that can put an array of temperature values in order from hottest to coldest. Sample output is given, along with a complete main program, and three function definitions. The last two function definitions on the following page are incomplete. Finish them, using the hints provided. Be extra careful with the order of parameters.

Sample output:

```
> ./a.out
Temps: 45.0 32.0 54.0 8.5 19.1
Hottest to Coldest: 54.0 45.0 32.0 19.1 8.5
>
```

Code:

```
/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */
/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}

void outputArrayOnOneLine(int count, double array[])
{
```

```

 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?

 About 3-4 lines of code is all you need.
 */
}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array

 About 6-8 lines of code will be enough. */

 int j;
 double temp;
 int indexOfColdest = 0; /* index of coldest element */

}

/* Esmerelda Weatherwax, CISC181, Fall 2004 TA:Jason Ogg */

```

```

/* Demonstrate call to function that puts temperatures in order */
/* from hottest to coldest */

#include <iostream>
using namespace std;

void hottestToColdest(int howMany, double temps[]);
void putColdestAtEnd(double tempArray[], int numTemps);
void outputArrayOnOneLine(int count, double array[]);

int main(void)
{
 double tempReadings[5]={45.0, 32.0, 54.0, 8.5, 19.1};

 /* output the temperatures before the function call */

 cout << "Temps: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 /* put temperatures in order */

 hottestToColdest(5, tempReadings);

 /* output the temperatures after the function call */

 cout << "Hottest to Coldest: ";
 outputArrayOnOneLine(5, tempReadings);
 cout << endl;

 return 0;
}

void outputArrayOnOneLine(int count, double array[])
{
 int i;
 cout << array[0];
 for (i=1; i<count; i++)
 cout << " " << array[i];
}

void hottestToColdest(int howMany, double temps[])
{
 /* Hints: use a for loop, and a call to putColdestAtEnd.

 For you to decide: does the call go inside or outside
 the for loop? Does the loop count up or down?
 What do you pass to putColdestAtEnd? Does it change
 or stay the same each time through the loop?
 */

 for (int i=howMany; i>1; i--)
 {
 putColdestAtEnd(temps, i);
 }
}

void putColdestAtEnd(double tempArray[], int numTemps)
{
 /* Hint: use a loop to find the index of coldest temperature,
 then swap that element with the last one in the array */

```

```

int j;
double temp;
int indexOfColdest = 0; /* index of coldest element */

for (j=1; j<numTemps; j++)
{
 if (tempArray[j] < tempArray[indexOfColdest])
 {
 indexOfColdest = j;
 }
}

temp = tempArray[indexOfColdest];
tempArray[indexOfColdest] = tempArray[numTemps - 1];
tempArray[numTemps - 1] = temp;
}

```

The following questions deal with this code excerpt:

```

// quiz question
#include <iostream>
using namespace std;

struct Point_S
{
 int x;
 int y;
};

int main(int argc, char*argv[])

{
 int a;
 int *b;
 int *c;

 double *d;
 double e;
 double f;
 double *g;

 Point_S *p;
 Point_S q;
 Point_S r;
 Point_S *s;
 Point_S *t;
}

```

```

b = new int;
c = &a;

g=new double
d=&e;

s = new Point_S;
t = &r;
p=&q;

cout << "Hi" << endl;
return 0;
}

```

22. (40 pts) Suppose you run the C++ program on the previous page, and the program reaches the line of code that prints out `Hi`.

Indicate whether, at that point in time, the expression given in the table on the left

- refers to a memory location on the *stack*,
- refers to a memory location on the *heap*,
- would cause an *error* (e.g. dereferencing something that isn't a pointer.)

Then, in the table on the right, fill in the type of each expression, or write "error" if the expression would not be valid (e.g. dereferencing something that isn't a pointer, or using the dot operator (`.`) on something that isn't a struct).

| <i>expression</i> | <i>Circle stack, heap or error</i> |
|-------------------|------------------------------------|
| a                 | stack heap error                   |
| *a                | stack heap error                   |
| b                 | stack heap error                   |
| *b                | stack heap error                   |
| c                 | stack heap error                   |
| *c                | stack heap error                   |
| *d                | stack heap error                   |
| *e                | stack heap error                   |
| f                 | stack heap error                   |
| *f                | stack heap error                   |
| g                 | stack heap error                   |
| *g                | stack heap error                   |
| p                 | stack heap error                   |
| *p                | stack heap error                   |
| *q                | stack heap error                   |
| *r                | stack heap error                   |
| *s                | stack heap error                   |
| *t                | stack heap error                   |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int         |
| *a                |             |
| &a                |             |
| b                 |             |
| *b                |             |
| &b                |             |
| d                 |             |
| *d                |             |
| e                 |             |
| &e                |             |
| p                 |             |
| *r                |             |
| &s                |             |
| t->x              |             |
| q.y               |             |
| q->y              |             |
| (*p).x            |             |
| &(p.x)            |             |
| argc              |             |
| argv[0]           |             |
| argv[0][1]        |             |
| argv              |             |
| &argc             |             |

| <i>expression</i> | <i>Answer</i> |
|-------------------|---------------|
| a                 | stack         |
| *a                | error         |
| b                 | stack         |
| *b                | heap          |
| c                 | stack         |
| *c                | stack         |
| *d                | stack         |
| *e                | error         |
| f                 | stack         |
| *f                | error         |
| g                 | stack         |
| *g                | heap          |
| p                 | stack         |
| *p                | stack         |
| *q                | error         |
| *r                | error         |
| *s                | heap          |
| *t                | stack         |

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a                 | int         |
| *a                | error       |
| &a                | int *       |
| b                 | int*        |
| *b                | int         |
| &b                | int**       |
| d                 | double *    |
| *d                | double      |
| e                 | double      |
| &e                | double *    |
| p                 | Point_S *   |
| *r                | error       |
| &s                | Point_S **  |
| t->x              | int         |
| q.y               | int         |
| q->y              | error       |
| (*p).x            | int         |
| &(p.x)            | error       |
| argc              | int         |
| argv[0]           | char *      |
| argv[0][1]        | char        |
| argv              | char **     |
| &argc             | int *       |

End of Key, seed 3456 version **D**  
Total Points: 100