

Name _____ A

UD Email address _____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

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!!!**

General Instructions

- The exam is 30% multiple choice, and 20% type expressions, and 50% “fill-in-the-blank” programming.
- The programming/short answer questions are on a separate sheet.
- Write your name **ONLY** in the spaces provided, so that the exam can be graded “name-blind”.
- You have 50 minutes. **Pace yourself**, and pay attention to the point values.
- Read *all* the directions *carefully* on each problem.
- Good luck.

1. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc
2 // P. Conrad for CISC181 exam
3
4 #include <iostream>
5 using namespace std;
6
7 void someFunction(const char *p)
8 {
9     char name[] = "fad";
10    cout << "p=" << p << endl;
11    p[0] = 'b';
12    p = name;
13 }
14
15 int main(void)
16 {
17     someFunction("foo");
18     return 0;
19 }
20
```

Which of the lines shown would produce a compiler error?

- (a) line 9
 - (b) line 10
 - (c) line 11
 - (d) line 12
2. (4 pts) C++ classes have a public part and a private part. This is most directly related to which aspect of Object-Oriented Programming?
- (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading

3. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc
2 // P. Conrad for CISC181 exam
3
4 #include <iostream>
5 using namespace std;
6
7 void someFunction(char * const p)
8 {
9     char name[] = "fad";
10    cout << "p=" << p << endl;
11    p[0] = 'b';
12    p = name;
13 }
14
15 int main(void)
16 {
17     char president[6]="Adams";
18     someFunction(president);
19     return 0;
20 }
21
```

Which of the lines shown would produce a compiler error?

- (a) line 9
- (b) line 10
- (c) line 11
- (d) line 12

4. (4 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+0`
- (c) `a++`
- (d) `++a`
- (e) `*a`

5. (4 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
6. (2 pts) Which of the following terms is commonly used to refer to member functions of a C++ class?
- (a) attributes
 - (b) parameters
 - (c) operands
 - (d) methods
7. (4 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?
- (a) `deref(p)`
 - (b) `*p`
 - (c) `&p`
 - (d) `%p`
 - (e) `$p`
8. (4 pts) Which of the following refers to the concept that C++ classes contain both data and functions in a single syntactic unit?
- (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading

9. (2 pts) Which of the following terms is commonly used to refer to data members of a C++ class?

- (a) attributes
- (b) parameters
- (c) operands
- (d) methods

Section 2. CISC181, E02, Spring 2006, Short Answer

Name _____ A

UD Email address _____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

Write your name and email ONLY in this spaces above.

It should appear nowhere else on this sheet of paper.

When finished, fold this page like a book with

- this as the “cover”
- question 10 on the inside
- question 11 on the “back”.

10. (50 pts) This sheet contains complete listings of four files, except that at a few places in the files, indicated by boxes and the labels (a) through (h), code has been removed.

Fill in the missing code.

Additional Information: These files **originally** contained complete correct source code for a C++ class specification, its member functions, a main program to test that class, and a Makefile to compile everything into an executable for the test program.

Each instance of this class represents a temperature reading taken in some room in Smith Hall. For example, an instance might represent the fact that the temperature in room 343 is 78.3 degrees.

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 // (a) [5 pts] Add something here to make this .h file idempotent
5
6
7
8 class TempReading_C
9 {
10 public:
11     TempReading_C(double theTemp, int theRoomNumber);
12
13     // (b) [5 pts] Fill in the function prototypes for the get functions
14
15
16
17
18     // rest of this file stays exactly as is
19
20     void setTemp(double theTemp) ;
21     void setRoomNumber(double theRoomNumber) ;
22
23 private:
24     double temp;
25     int roomNumber;
26 };
27
28 #endif // TEMPREADING_H
```

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8     temp = theTemp;
9     roomNumber = theRoomNumber;
10 }
11
12 // (c) [10 pts] fill in the entire function definitions for the get functions
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27 void TempReading_C::setTemp(double theTemp)
28 {
29     // (d) [4 pts] In this functions, just fill in the body
30
31
32 }
33
34 void TempReading_C::setRoomNumber(double theRoomNumber)
35 {
36     // (e) [4 pts] In this function, just fill in the body
37
38
39
40 }
```

```

1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3 #include <iostream>
4 using std::cout;
5 using std::cerr;
6 using std::endl;
7
8 // (f) [4 pts] Something necessary is missing here
9 // Without it you will get the error
10 // Error: TempReading_C is not defined.
11
12
13
14 void testSetterGetter(double temp, int room )
15 {
16     TempReading_C t(0.0, 0);
17
18     // (g) [4 pts] Fill in code here to invoke the two setter methods
19     // on the object t, setting the temperature and room number
20     // from the values passed in to the function testSetterGetter
21
22
23     // Test whether the setter and getter methods are
24     // working correctly.
25
26     if (t.getTemp() == temp && t.getRoomNumber() == room)
27         cout << "Passed";
28     else
29         cout << "Failed";
30 }
31
32
33 void testConstructor(double temp, int room )
34 {
35
36     // (h) [4 pts] Fill in one line of code to invoke the constructor
37     // in such a way that the test below makes sense.
38
39
40
41     // Test whether the setter and getter methods are
42     // working correctly.
43
44     if (x.getTemp() == temp && x.getRoomNumber() == room)
45         cout << "Passed";
46     else
47         cout << "Failed";
48 }
49
50
51 int main(void)
52 {
53     testSetterGetter(78.3, 343);
54     testSetterGetter(81.2, 326);
55     testSetterGetter(72.1, 101);
56
57     testConstructor(78.3, 343);
58     testConstructor(81.2, 326);
59     testConstructor(72.1, 101);
60
61     return(0);
62 }

```

```
1 # Makefile for tempReading class
2 # Complete except for one line
3
4 # (i) [5 pts] The line starting with CCC= is incomplete. Finish it.
5 # There are two possible answers.
6
7 CCC =
8
9 BINARIES = testTempReading
10
11 all: ${BINARIES}
12
13 # (j) [5 pts] The line below containing ${CCC} is incomplete; finish it!
14
15 testTempReading: testTempReading.o tempReading.o
16     ${CCC}
17
18 clean:
19     /bin/rm -f *.o a.out core ${BINARIES}
20
21
22
```

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;
}
```

11. (20 pts) Suppose you run the C++ program shown, and the program reaches the line of code that prints out Hi.

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

End of Exam. Total Points: 100

Key: version **A** CISC 181 sections 010-015, Midterm 2 04/19/06

1. (c)
2. (b)
3. (d)
4. (e)
5. (b)
6. (d)
7. (b)
8. (a)
9. (a)

Section 2. CISC181, E02, Spring 2006, Short Answer

```
10.1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #ifndef TEMPREADING_H
5 #define TEMPREADING_H
6
7 class TempReading_C
8 {
9 public:
10 TempReading_C(double theTemp, int theRoomNumber);
11
12 double getTemp() const;
13 int getRoomNumber() const;
14
15 void setTemp(double theTemp) ;
16 void setRoomNumber(double theRoomNumber) ;
17
18 private:
19 double temp;
20 int roomNumber;
21 };
22
23 #endif // TEMPREADING_H
```

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8 temp = theTemp;
9 roomNumber = theRoomNumber;
10 }
11
12 double TempReading_C::getTemp() const
13 {
14 return temp;
15 }
16
17 int TempReading_C::getRoomNumber() const
18 {
19 return roomNumber;
20 }
21
22 void TempReading_C::setTemp(double theTemp)
23 {
24 temp = theTemp;
25 }
26
27 void TempReading_C::setRoomNumber(double theRoomNumber)
28 {
29 roomNumber = theRoomNumber;
30 }
31 }
```

```
1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3
4 #include <iostream>
5 using std::cout;
6 using std::cerr;
```

```

7 using std::endl;
8
9 #include "tempReading.h"
10
11 void testSetterGetter(double temp, int room )
12 {
13
14     TempReading_C t(0.0, 0);
15
16     t.setTemp(temp); t.setRoomNumber(room);
17
18     if (t.getTemp() == temp && t.getRoomNumber() == room)
19         cout << "Passed";
20     else
21         cout << "Failed";
22
23 }
24
25
26
27 void testConstructor(double temp, int room )
28 {
29     TempReading_C x(temp, room);
30
31
32     if (x.getTemp() == temp && x.getRoomNumber() == room)
33         cout << "Passed";
34     else
35         cout << "Failed";
36
37 }
38
39
40 int main(void)
41 {
42
43     testSetterGetter(78.3, 343);
44     testSetterGetter(81.2, 326);
45     testSetterGetter(72.1, 101);
46
47     testConstructor(78.3, 343);
48     testConstructor(81.2, 326);
49     testConstructor(72.1, 101);
50
51     return(0);
52 }

```

```

1 # Makefile for tempReading class
2
3 CCC = CC
4
5 BINARIES = testTempReading
6
7 all: ${BINARIES}
8
9 testTempReading: testTempReading.o tempReading.o
10     ${CCC} testTempReading.o tempReading.o -o $@
11
12 clean:
13     /bin/rm -f *.o a.out core ${BINARIES}
14
15
16

```

11.

| <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 * |

Name _____ B

UD Email address _____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

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!!!**

General Instructions

- The exam is 30% multiple choice, and 20% type expressions, and 50% “fill-in-the-blank” programming.
- The programming/short answer questions are on a separate sheet.
- Write your name **ONLY** in the spaces provided, so that the exam can be graded “name-blind”.
- You have 50 minutes. **Pace yourself**, and pay attention to the point values.
- Read *all* the directions *carefully* on each problem.
- Good luck.

1. (4 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?
 - (a) `deref(p)`
 - (b) `*p`
 - (c) `&p`
 - (d) `%p`
 - (e) `$p`
2. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc
2 // P. Conrad for CISC181 exam
3
4 #include <iostream>
5 using namespace std;
6
7 void someFunction(const char *p)
8 {
9     char name[] = "fad";
10    cout << "p=" << p << endl;
11    p[0] = 'b';
12    p = name;
13 }
14
15 int main(void)
16 {
17     someFunction("foo");
18     return 0;
19 }
20
```

Which of the lines shown would produce a compiler error?

- (a) line 9
- (b) line 10
- (c) line 11
- (d) line 12

3. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc
2 // P. Conrad for CISC181 exam
3
4 #include <iostream>
5 using namespace std;
6
7 void someFunction(char * const p)
8 {
9     char name[] = "fad";
10    cout << "p=" << p << endl;
11    p[0] = 'b';
12    p = name;
13 }
14
15 int main(void)
16 {
17     char president[6]="Adams";
18     someFunction(president);
19     return 0;
20 }
21
```

Which of the lines shown would produce a compiler error?

- (a) line 9
 - (b) line 10
 - (c) line 11
 - (d) line 12
4. (2 pts) Which of the following terms is commonly used to refer to member functions of a C++ class?
- (a) attributes
 - (b) parameters
 - (c) operands
 - (d) methods
5. (4 pts) C++ classes have a public part and a private part. This is most directly related to which aspect of Object-Oriented Programming?
- (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading

6. (2 pts) Which of the following terms is commonly used to refer to data members of a C++ class?
- (a) attributes
 - (b) parameters
 - (c) operands
 - (d) methods
7. (4 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
8. (4 pts) Which of the following refers to the concept that C++ classes contain both data and functions in a single syntactic unit?
- (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading
9. (4 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+0`
 - (c) `a++`
 - (d) `++a`
 - (e) `*a`

Section 2. CISC181, E02, Spring 2006, Short Answer

Name_____ B

UD Email address_____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

Write your name and email ONLY in this spaces above.

It should appear nowhere else on this sheet of paper.

When finished, fold this page like a book with

- this as the “cover”
- question 10 on the inside
- question 11 on the “back”.

10. (50 pts) This sheet contains complete listings of four files, except that at a few places in the files, indicated by boxes and the labels (a) through (h), code has been removed.

Fill in the missing code.

Additional Information: These files **originally** contained complete correct source code for a C++ class specification, its member functions, a main program to test that class, and a Makefile to compile everything into an executable for the test program.

Each instance of this class represents a temperature reading taken in some room in Smith Hall. For example, an instance might represent the fact that the temperature in room 343 is 78.3 degrees.

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 // (a) [5 pts] Add something here to make this .h file idempotent
5
6
7
8 class TempReading_C
9 {
10 public:
11     TempReading_C(double theTemp, int theRoomNumber);
12
13     // (b) [5 pts] Fill in the function prototypes for the get functions
14
15
16
17
18     // rest of this file stays exactly as is
19
20     void setTemp(double theTemp) ;
21     void setRoomNumber(double theRoomNumber) ;
22
23 private:
24     double temp;
25     int roomNumber;
26 };
27
28 #endif // TEMPREADING_H
```

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8     temp = theTemp;
9     roomNumber = theRoomNumber;
10 }
11
12 // (c) [10 pts] fill in the entire function definitions for the get functions
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27 void TempReading_C::setTemp(double theTemp)
28 {
29     // (d) [4 pts] In this functions, just fill in the body
30
31
32 }
33
34 void TempReading_C::setRoomNumber(double theRoomNumber)
35 {
36     // (e) [4 pts] In this function, just fill in the body
37
38
39
40 }
```

```

1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3 #include <iostream>
4 using std::cout;
5 using std::cerr;
6 using std::endl;
7
8 // (f) [4 pts] Something necessary is missing here
9 // Without it you will get the error
10 // Error: TempReading_C is not defined.
11
12
13
14 void testSetterGetter(double temp, int room )
15 {
16     TempReading_C t(0.0, 0);
17
18     // (g) [4 pts] Fill in code here to invoke the two setter methods
19     // on the object t, setting the temperature and room number
20     // from the values passed in to the function testSetterGetter
21
22
23     // Test whether the setter and getter methods are
24     // working correctly.
25
26     if (t.getTemp() == temp && t.getRoomNumber() == room)
27         cout << "Passed";
28     else
29         cout << "Failed";
30 }
31
32
33 void testConstructor(double temp, int room )
34 {
35
36     // (h) [4 pts] Fill in one line of code to invoke the constructor
37     // in such a way that the test below makes sense.
38
39
40
41     // Test whether the setter and getter methods are
42     // working correctly.
43
44     if (x.getTemp() == temp && x.getRoomNumber() == room)
45         cout << "Passed";
46     else
47         cout << "Failed";
48 }
49
50
51 int main(void)
52 {
53     testSetterGetter(78.3, 343);
54     testSetterGetter(81.2, 326);
55     testSetterGetter(72.1, 101);
56
57     testConstructor(78.3, 343);
58     testConstructor(81.2, 326);
59     testConstructor(72.1, 101);
60
61     return(0);
62 }

```

```
1 # Makefile for tempReading class
2 # Complete except for one line
3
4 # (i) [5 pts] The line starting with CCC= is incomplete. Finish it.
5 # There are two possible answers.
6
7 CCC =
8
9 BINARIES = testTempReading
10
11 all: ${BINARIES}
12
13 # (j) [5 pts] The line below containing ${CCC} is incomplete; finish it!
14
15 testTempReading: testTempReading.o tempReading.o
16     ${CCC}
17
18 clean:
19     /bin/rm -f *.o a.out core ${BINARIES}
20
21
22
```

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;
}
```

11. (20 pts) Suppose you run the C++ program shown, and the program reaches the line of code that prints out Hi.

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

End of Exam. Total Points: 100

Key: version **B** CISC 181 sections 010-015, Midterm 2 04/19/06

1. (b)
2. (c)
3. (d)
4. (d)
5. (b)
6. (a)
7. (b)
8. (a)
9. (e)

Section 2. CISC181, E02, Spring 2006, Short Answer

```
10.1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #ifndef TEMPREADING_H
5 #define TEMPREADING_H
6
7 class TempReading_C
8 {
9 public:
10 TempReading_C(double theTemp, int theRoomNumber);
11
12 double getTemp() const;
13 int getRoomNumber() const;
14
15 void setTemp(double theTemp) ;
16 void setRoomNumber(double theRoomNumber) ;
17
18 private:
19 double temp;
20 int roomNumber;
21 };
22
23 #endif // TEMPREADING_H
```

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8     temp = theTemp;
9     roomNumber = theRoomNumber;
10 }
11
12 double TempReading_C::getTemp() const
13 {
14     return temp;
15 }
16
17 int TempReading_C::getRoomNumber() const
18 {
19     return roomNumber;
20 }
21
22 void TempReading_C::setTemp(double theTemp)
23 {
24     temp = theTemp;
25 }
26
27 void TempReading_C::setRoomNumber(double theRoomNumber)
28 {
29     roomNumber = theRoomNumber;
30 }
31 }
```

```
1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3
4 #include <iostream>
5 using std::cout;
6 using std::cerr;
```

```

7 using std::endl;
8
9 #include "tempReading.h"
10
11 void testSetterGetter(double temp, int room )
12 {
13
14     TempReading_C t(0.0, 0);
15
16     t.setTemp(temp); t.setRoomNumber(room);
17
18     if (t.getTemp() == temp && t.getRoomNumber() == room)
19         cout << "Passed";
20     else
21         cout << "Failed";
22
23 }
24
25
26
27 void testConstructor(double temp, int room )
28 {
29     TempReading_C x(temp, room);
30
31
32     if (x.getTemp() == temp && x.getRoomNumber() == room)
33         cout << "Passed";
34     else
35         cout << "Failed";
36
37 }
38
39
40 int main(void)
41 {
42
43     testSetterGetter(78.3, 343);
44     testSetterGetter(81.2, 326);
45     testSetterGetter(72.1, 101);
46
47     testConstructor(78.3, 343);
48     testConstructor(81.2, 326);
49     testConstructor(72.1, 101);
50
51     return(0);
52 }

```

```

1 # Makefile for tempReading class
2
3 CCC = CC
4
5 BINARIES = testTempReading
6
7 all: ${BINARIES}
8
9 testTempReading: testTempReading.o tempReading.o
10     ${CCC} testTempReading.o tempReading.o -o $@
11
12 clean:
13     /bin/rm -f *.o a.out core ${BINARIES}
14
15
16

```

11.

| <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 * |

Name _____

UD Email address _____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

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!!!**

General Instructions

- The exam is 30% multiple choice, and 20% type expressions, and 50% “fill-in-the-blank” programming.
- The programming/short answer questions are on a separate sheet.
- Write your name **ONLY** in the spaces provided, so that the exam can be graded “name-blind”.
- You have 50 minutes. **Pace yourself**, and pay attention to the point values.
- Read *all* the directions *carefully* on each problem.
- Good luck.

1. (4 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+0`
 - (c) `a++`
 - (d) `++a`
 - (e) `*a`

2. (4 pts) C++ classes have a public part and a private part. This is most directly related to which aspect of Object-Oriented Programming?
 - (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading

3. (4 pts) Which of the following refers to the concept that C++ classes contain both data and functions in a single syntactic unit?
 - (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading

4. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc
2 // P. Conrad for CISC181 exam
3
4 #include <iostream>
5 using namespace std;
6
7 void someFunction(char * const p)
8 {
9     char name[] = "fad";
10    cout << "p=" << p << endl;
11    p[0] = 'b';
12    p = name;
13 }
14
15 int main(void)
16 {
17     char president[6]="Adams";
18     someFunction(president);
19     return 0;
20 }
21
```

Which of the lines shown would produce a compiler error?

- (a) line 9
- (b) line 10
- (c) line 11
- (d) line 12

5. (4 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?

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

6. (4 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
7. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc  
2 // P. Conrad for CISC181 exam  
3  
4 #include <iostream>  
5 using namespace std;  
6  
7 void someFunction(const char *p)  
8 {  
9     char name[] = "fad";  
10    cout << "p=" << p << endl;  
11    p[0] = 'b';  
12    p = name;  
13 }  
14  
15 int main(void)  
16 {  
17     someFunction("foo");  
18     return 0;  
19 }  
20
```

Which of the lines shown would produce a compiler error?

- (a) line 9
- (b) line 10
- (c) line 11
- (d) line 12

8. (2 pts) Which of the following terms is commonly used to refer to data members of a C++ class?
- (a) attributes
 - (b) parameters
 - (c) operands
 - (d) methods
9. (2 pts) Which of the following terms is commonly used to refer to member functions of a C++ class?
- (a) attributes
 - (b) parameters
 - (c) operands
 - (d) methods

Section 2. CISC181, E02, Spring 2006, Short Answer

Name _____

UD Email address _____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

Write your name and email ONLY in this spaces above.

It should appear nowhere else on this sheet of paper.

When finished, fold this page like a book with

- this as the “cover”
- question 10 on the inside
- question 11 on the “back”.

10. (50 pts) This sheet contains complete listings of four files, except that at a few places in the files, indicated by boxes and the labels (a) through (h), code has been removed.

Fill in the missing code.

Additional Information: These files **originally** contained complete correct source code for a C++ class specification, its member functions, a main program to test that class, and a Makefile to compile everything into an executable for the test program.

Each instance of this class represents a temperature reading taken in some room in Smith Hall. For example, an instance might represent the fact that the temperature in room 343 is 78.3 degrees.

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 // (a) [5 pts] Add something here to make this .h file idempotent
5
6
7
8 class TempReading_C
9 {
10 public:
11     TempReading_C(double theTemp, int theRoomNumber);
12
13     // (b) [5 pts] Fill in the function prototypes for the get functions
14
15
16
17
18     // rest of this file stays exactly as is
19
20     void setTemp(double theTemp) ;
21     void setRoomNumber(double theRoomNumber) ;
22
23 private:
24     double temp;
25     int roomNumber;
26 };
27
28 #endif // TEMPREADING_H
```

```

1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8     temp = theTemp;
9     roomNumber = theRoomNumber;
10 }
11
12 // (c) [10 pts] fill in the entire function definitions for the get functions
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27 void TempReading_C::setTemp(double theTemp)
28 {
29     // (d) [4 pts] In this functions, just fill in the body
30
31
32 }
33
34 void TempReading_C::setRoomNumber(double theRoomNumber)
35 {
36     // (e) [4 pts] In this function, just fill in the body
37
38
39
40 }

```

```

1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3 #include <iostream>
4 using std::cout;
5 using std::cerr;
6 using std::endl;
7
8 // (f) [4 pts] Something necessary is missing here
9 // Without it you will get the error
10 // Error: TempReading_C is not defined.
11
12
13
14 void testSetterGetter(double temp, int room )
15 {
16     TempReading_C t(0.0, 0);
17
18     // (g) [4 pts] Fill in code here to invoke the two setter methods
19     // on the object t, setting the temperature and room number
20     // from the values passed in to the function testSetterGetter
21
22
23     // Test whether the setter and getter methods are
24     // working correctly.
25
26     if (t.getTemp() == temp && t.getRoomNumber() == room)
27         cout << "Passed";
28     else
29         cout << "Failed";
30 }
31
32
33 void testConstructor(double temp, int room )
34 {
35
36     // (h) [4 pts] Fill in one line of code to invoke the constructor
37     // in such a way that the test below makes sense.
38
39
40
41     // Test whether the setter and getter methods are
42     // working correctly.
43
44     if (x.getTemp() == temp && x.getRoomNumber() == room)
45         cout << "Passed";
46     else
47         cout << "Failed";
48 }
49
50
51 int main(void)
52 {
53     testSetterGetter(78.3, 343);
54     testSetterGetter(81.2, 326);
55     testSetterGetter(72.1, 101);
56
57     testConstructor(78.3, 343);
58     testConstructor(81.2, 326);
59     testConstructor(72.1, 101);
60
61     return(0);
62 }

```

```
1 # Makefile for tempReading class
2 # Complete except for one line
3
4 # (i) [5 pts] The line starting with CCC= is incomplete. Finish it.
5 # There are two possible answers.
6
7 CCC =
8
9 BINARIES = testTempReading
10
11 all: ${BINARIES}
12
13 # (j) [5 pts] The line below containing ${CCC} is incomplete; finish it!
14
15 testTempReading: testTempReading.o tempReading.o
16     ${CCC}
17
18 clean:
19     /bin/rm -f *.o a.out core ${BINARIES}
20
21
22
```

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;
}
```

11. (20 pts) Suppose you run the C++ program shown, and the program reaches the line of code that prints out Hi.

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

End of Exam. Total Points: 100

Key: version **C** CISC 181 sections 010-015, Midterm 2 04/19/06

1. (e)
2. (b)
3. (a)
4. (d)
5. (b)
6. (b)
7. (c)
8. (a)
9. (d)

Section 2. CISC181, E02, Spring 2006, Short Answer

```
10.1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #ifndef TEMPREADING_H
5 #define TEMPREADING_H
6
7 class TempReading_C
8 {
9 public:
10 TempReading_C(double theTemp, int theRoomNumber);
11
12 double getTemp() const;
13 int getRoomNumber() const;
14
15 void setTemp(double theTemp) ;
16 void setRoomNumber(double theRoomNumber) ;
17
18 private:
19 double temp;
20 int roomNumber;
21 };
22
23 #endif // TEMPREADING_H
```

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8 temp = theTemp;
9 roomNumber = theRoomNumber;
10 }
11
12 double TempReading_C::getTemp() const
13 {
14 return temp;
15 }
16
17 int TempReading_C::getRoomNumber() const
18 {
19 return roomNumber;
20 }
21
22 void TempReading_C::setTemp(double theTemp)
23 {
24 temp = theTemp;
25 }
26
27 void TempReading_C::setRoomNumber(double theRoomNumber)
28 {
29 roomNumber = theRoomNumber;
30 }
31 }
```

```
1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3
4 #include <iostream>
5 using std::cout;
6 using std::cerr;
```

```

7 using std::endl;
8
9 #include "tempReading.h"
10
11 void testSetterGetter(double temp, int room )
12 {
13
14     TempReading_C t(0.0, 0);
15
16     t.setTemp(temp); t.setRoomNumber(room);
17
18     if (t.getTemp() == temp && t.getRoomNumber() == room)
19         cout << "Passed";
20     else
21         cout << "Failed";
22
23 }
24
25
26
27 void testConstructor(double temp, int room )
28 {
29     TempReading_C x(temp, room);
30
31
32     if (x.getTemp() == temp && x.getRoomNumber() == room)
33         cout << "Passed";
34     else
35         cout << "Failed";
36
37 }
38
39
40 int main(void)
41 {
42
43     testSetterGetter(78.3, 343);
44     testSetterGetter(81.2, 326);
45     testSetterGetter(72.1, 101);
46
47     testConstructor(78.3, 343);
48     testConstructor(81.2, 326);
49     testConstructor(72.1, 101);
50
51     return(0);
52 }

```

```

1 # Makefile for tempReading class
2
3 CCC = CC
4
5 BINARIES = testTempReading
6
7 all: ${BINARIES}
8
9 testTempReading: testTempReading.o tempReading.o
10     ${CCC} testTempReading.o tempReading.o -o $@
11
12 clean:
13     /bin/rm -f *.o a.out core ${BINARIES}
14
15
16

```

11.

| <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 * |

Name _____ D

UD Email address _____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

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!!!**

General Instructions

- The exam is 30% multiple choice, and 20% type expressions, and 50% “fill-in-the-blank” programming.
- The programming/short answer questions are on a separate sheet.
- Write your name **ONLY** in the spaces provided, so that the exam can be graded “name-blind”.
- You have 50 minutes. **Pace yourself**, and pay attention to the point values.
- Read *all* the directions *carefully* on each problem.
- Good luck.

1. (2 pts) Which of the following terms is commonly used to refer to data members of a C++ class?
 - (a) attributes
 - (b) parameters
 - (c) operands
 - (d) methods

2. (4 pts) C++ classes have a public part and a private part. This is most directly related to which aspect of Object-Oriented Programming?
 - (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading

3. (4 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

4. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc
2 // P. Conrad for CISC181 exam
3
4 #include <iostream>
5 using namespace std;
6
7 void someFunction(char * const p)
8 {
9     char name[] = "fad";
10    cout << "p=" << p << endl;
11    p[0] = 'b';
12    p = name;
13 }
14
15 int main(void)
16 {
17     char president[6]="Adams";
18     someFunction(president);
19     return 0;
20 }
21
```

Which of the lines shown would produce a compiler error?

- (a) line 9
 - (b) line 10
 - (c) line 11
 - (d) line 12
5. (4 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?
- (a) `deref(p)`
 - (b) `*p`
 - (c) `&p`
 - (d) `%p`
 - (e) `$p`
6. (4 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+0`
 - (c) `a++`
 - (d) `++a`
 - (e) `*a`

7. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc
2 // P. Conrad for CISC181 exam
3
4 #include <iostream>
5 using namespace std;
6
7 void someFunction(const char *p)
8 {
9     char name[] = "fad";
10    cout << "p=" << p << endl;
11    p[0] = 'b';
12    p = name;
13 }
14
15 int main(void)
16 {
17     someFunction("foo");
18     return 0;
19 }
20
```

Which of the lines shown would produce a compiler error?

- (a) line 9
 - (b) line 10
 - (c) line 11
 - (d) line 12
8. (4 pts) Which of the following refers to the concept that C++ classes contain both data and functions in a single syntactic unit?
- (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading
9. (2 pts) Which of the following terms is commonly used to refer to member functions of a C++ class?
- (a) attributes
 - (b) parameters
 - (c) operands
 - (d) methods

Section 2. CISC181, E02, Spring 2006, Short Answer

Name_____ D

UD Email address_____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

Write your name and email ONLY in this spaces above.

It should appear nowhere else on this sheet of paper.

When finished, fold this page like a book with

- this as the “cover”
- question 10 on the inside
- question 11 on the “back”.

10. (50 pts) This sheet contains complete listings of four files, except that at a few places in the files, indicated by boxes and the labels (a) through (h), code has been removed.

Fill in the missing code.

Additional Information: These files **originally** contained complete correct source code for a C++ class specification, its member functions, a main program to test that class, and a Makefile to compile everything into an executable for the test program.

Each instance of this class represents a temperature reading taken in some room in Smith Hall. For example, an instance might represent the fact that the temperature in room 343 is 78.3 degrees.

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 // (a) [5 pts] Add something here to make this .h file idempotent
5
6
7
8 class TempReading_C
9 {
10 public:
11     TempReading_C(double theTemp, int theRoomNumber);
12
13     // (b) [5 pts] Fill in the function prototypes for the get functions
14
15
16
17
18     // rest of this file stays exactly as is
19
20     void setTemp(double theTemp) ;
21     void setRoomNumber(double theRoomNumber) ;
22
23 private:
24     double temp;
25     int roomNumber;
26 };
27
28 #endif // TEMPREADING_H
```

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8     temp = theTemp;
9     roomNumber = theRoomNumber;
10 }
11
12 // (c) [10 pts] fill in the entire function definitions for the get functions
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27 void TempReading_C::setTemp(double theTemp)
28 {
29     // (d) [4 pts] In this functions, just fill in the body
30
31
32 }
33
34 void TempReading_C::setRoomNumber(double theRoomNumber)
35 {
36     // (e) [4 pts] In this function, just fill in the body
37
38
39
40 }
```

```

1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3 #include <iostream>
4 using std::cout;
5 using std::cerr;
6 using std::endl;
7
8 // (f) [4 pts] Something necessary is missing here
9 // Without it you will get the error
10 // Error: TempReading_C is not defined.
11
12
13
14 void testSetterGetter(double temp, int room )
15 {
16     TempReading_C t(0.0, 0);
17
18     // (g) [4 pts] Fill in code here to invoke the two setter methods
19     // on the object t, setting the temperature and room number
20     // from the values passed in to the function testSetterGetter
21
22
23     // Test whether the setter and getter methods are
24     // working correctly.
25
26     if (t.getTemp() == temp && t.getRoomNumber() == room)
27         cout << "Passed";
28     else
29         cout << "Failed";
30 }
31
32
33 void testConstructor(double temp, int room )
34 {
35
36     // (h) [4 pts] Fill in one line of code to invoke the constructor
37     // in such a way that the test below makes sense.
38
39
40
41     // Test whether the setter and getter methods are
42     // working correctly.
43
44     if (x.getTemp() == temp && x.getRoomNumber() == room)
45         cout << "Passed";
46     else
47         cout << "Failed";
48 }
49
50
51 int main(void)
52 {
53     testSetterGetter(78.3, 343);
54     testSetterGetter(81.2, 326);
55     testSetterGetter(72.1, 101);
56
57     testConstructor(78.3, 343);
58     testConstructor(81.2, 326);
59     testConstructor(72.1, 101);
60
61     return(0);
62 }

```

```
1 # Makefile for tempReading class
2 # Complete except for one line
3
4 # (i) [5 pts] The line starting with CCC= is incomplete. Finish it.
5 # There are two possible answers.
6
7 CCC =
8
9 BINARIES = testTempReading
10
11 all: ${BINARIES}
12
13 # (j) [5 pts] The line below containing ${CCC} is incomplete; finish it!
14
15 testTempReading: testTempReading.o tempReading.o
16     ${CCC}
17
18 clean:
19     /bin/rm -f *.o a.out core ${BINARIES}
20
21
22
```

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;
}
```

11. (20 pts) Suppose you run the C++ program shown, and the program reaches the line of code that prints out Hi.

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

End of Exam. Total Points: 100

Key: version **D** CISC 181 sections 010-015, Midterm 2 04/19/06

1. (a)
2. (b)
3. (b)
4. (d)
5. (b)
6. (e)
7. (c)
8. (a)
9. (d)

Section 2. CISC181, E02, Spring 2006, Short Answer

```
10.1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #ifndef TEMPREADING_H
5 #define TEMPREADING_H
6
7 class TempReading_C
8 {
9 public:
10 TempReading_C(double theTemp, int theRoomNumber);
11
12 double getTemp() const;
13 int getRoomNumber() const;
14
15 void setTemp(double theTemp) ;
16 void setRoomNumber(double theRoomNumber) ;
17
18 private:
19 double temp;
20 int roomNumber;
21 };
22
23 #endif // TEMPREADING_H
```

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8 temp = theTemp;
9 roomNumber = theRoomNumber;
10 }
11
12 double TempReading_C::getTemp() const
13 {
14 return temp;
15 }
16
17 int TempReading_C::getRoomNumber() const
18 {
19 return roomNumber;
20 }
21
22 void TempReading_C::setTemp(double theTemp)
23 {
24 temp = theTemp;
25 }
26
27 void TempReading_C::setRoomNumber(double theRoomNumber)
28 {
29 roomNumber = theRoomNumber;
30 }
31 }
```

```
1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3
4 #include <iostream>
5 using std::cout;
6 using std::cerr;
```

```

7 using std::endl;
8
9 #include "tempReading.h"
10
11 void testSetterGetter(double temp, int room )
12 {
13
14     TempReading_C t(0.0, 0);
15
16     t.setTemp(temp); t.setRoomNumber(room);
17
18     if (t.getTemp() == temp && t.getRoomNumber() == room)
19         cout << "Passed";
20     else
21         cout << "Failed";
22
23 }
24
25
26
27 void testConstructor(double temp, int room )
28 {
29     TempReading_C x(temp, room);
30
31
32     if (x.getTemp() == temp && x.getRoomNumber() == room)
33         cout << "Passed";
34     else
35         cout << "Failed";
36
37 }
38
39
40 int main(void)
41 {
42
43     testSetterGetter(78.3, 343);
44     testSetterGetter(81.2, 326);
45     testSetterGetter(72.1, 101);
46
47     testConstructor(78.3, 343);
48     testConstructor(81.2, 326);
49     testConstructor(72.1, 101);
50
51     return(0);
52 }

```

```

1 # Makefile for tempReading class
2
3 CCC = CC
4
5 BINARIES = testTempReading
6
7 all: ${BINARIES}
8
9 testTempReading: testTempReading.o tempReading.o
10     ${CCC} testTempReading.o tempReading.o -o $@
11
12 clean:
13     /bin/rm -f *.o a.out core ${BINARIES}
14
15
16

```

11.

| <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, version **D** Total Points: 0

Name _____ E

UD Email address _____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

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!!!**

General Instructions

- The exam is 30% multiple choice, and 20% type expressions, and 50% “fill-in-the-blank” programming.
- The programming/short answer questions are on a separate sheet.
- Write your name **ONLY** in the spaces provided, so that the exam can be graded “name-blind”.
- You have 50 minutes. **Pace yourself**, and pay attention to the point values.
- Read *all* the directions *carefully* on each problem.
- Good luck.

1. (4 pts) Which of the following refers to the concept that C++ classes contain both data and functions in a single syntactic unit?
 - (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading
2. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc
2 // P. Conrad for CISC181 exam
3
4 #include <iostream>
5 using namespace std;
6
7 void someFunction(char * const p)
8 {
9     char name[] = "fad";
10    cout << "p=" << p << endl;
11    p[0] = 'b';
12    p = name;
13 }
14
15 int main(void)
16 {
17     char president[6]="Adams";
18     someFunction(president);
19     return 0;
20 }
21
```

Which of the lines shown would produce a compiler error?

- (a) line 9
 - (b) line 10
 - (c) line 11
 - (d) line 12
3. (2 pts) Which of the following terms is commonly used to refer to data members of a C++ class?
 - (a) attributes
 - (b) parameters
 - (c) operands
 - (d) methods

4. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc
2 // P. Conrad for CISC181 exam
3
4 #include <iostream>
5 using namespace std;
6
7 void someFunction(const char *p)
8 {
9     char name[] = "fad";
10    cout << "p=" << p << endl;
11    p[0] = 'b';
12    p = name;
13 }
14
15 int main(void)
16 {
17     someFunction("foo");
18     return 0;
19 }
20
```

Which of the lines shown would produce a compiler error?

- (a) line 9
- (b) line 10
- (c) line 11
- (d) line 12

5. (2 pts) Which of the following terms is commonly used to refer to member functions of a C++ class?

- (a) attributes
- (b) parameters
- (c) operands
- (d) methods

6. (4 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
7. (4 pts) C++ classes have a public part and a private part. This is most directly related to which aspect of Object-Oriented Programming?
- (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading
8. (4 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?
- (a) `deref(p)`
 - (b) `*p`
 - (c) `&p`
 - (d) `%p`
 - (e) `$p`
9. (4 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+0`
 - (c) `a++`
 - (d) `++a`
 - (e) `*a`

Section 2. CISC181, E02, Spring 2006, Short Answer

Name _____ E

UD Email address _____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

Write your name and email ONLY in this spaces above.

It should appear nowhere else on this sheet of paper.

When finished, fold this page like a book with

- this as the “cover”
- question 10 on the inside
- question 11 on the “back”.

10. (50 pts) This sheet contains complete listings of four files, except that at a few places in the files, indicated by boxes and the labels (a) through (h), code has been removed.

Fill in the missing code.

Additional Information: These files **originally** contained complete correct source code for a C++ class specification, its member functions, a main program to test that class, and a Makefile to compile everything into an executable for the test program.

Each instance of this class represents a temperature reading taken in some room in Smith Hall. For example, an instance might represent the fact that the temperature in room 343 is 78.3 degrees.

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 // (a) [5 pts] Add something here to make this .h file idempotent
5
6
7
8 class TempReading_C
9 {
10 public:
11     TempReading_C(double theTemp, int theRoomNumber);
12
13     // (b) [5 pts] Fill in the function prototypes for the get functions
14
15
16
17
18     // rest of this file stays exactly as is
19
20     void setTemp(double theTemp) ;
21     void setRoomNumber(double theRoomNumber) ;
22
23 private:
24     double temp;
25     int roomNumber;
26 };
27
28 #endif // TEMPREADING_H
```

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8     temp = theTemp;
9     roomNumber = theRoomNumber;
10 }
11
12 // (c) [10 pts] fill in the entire function definitions for the get functions
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27 void TempReading_C::setTemp(double theTemp)
28 {
29     // (d) [4 pts] In this functions, just fill in the body
30
31
32 }
33
34 void TempReading_C::setRoomNumber(double theRoomNumber)
35 {
36     // (e) [4 pts] In this function, just fill in the body
37
38
39
40 }
```

```

1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3 #include <iostream>
4 using std::cout;
5 using std::cerr;
6 using std::endl;
7
8 // (f) [4 pts] Something necessary is missing here
9 // Without it you will get the error
10 // Error: TempReading_C is not defined.
11
12
13
14 void testSetterGetter(double temp, int room )
15 {
16     TempReading_C t(0.0, 0);
17
18     // (g) [4 pts] Fill in code here to invoke the two setter methods
19     // on the object t, setting the temperature and room number
20     // from the values passed in to the function testSetterGetter
21
22
23     // Test whether the setter and getter methods are
24     // working correctly.
25
26     if (t.getTemp() == temp && t.getRoomNumber() == room)
27         cout << "Passed";
28     else
29         cout << "Failed";
30 }
31
32
33 void testConstructor(double temp, int room )
34 {
35
36     // (h) [4 pts] Fill in one line of code to invoke the constructor
37     // in such a way that the test below makes sense.
38
39
40
41     // Test whether the setter and getter methods are
42     // working correctly.
43
44     if (x.getTemp() == temp && x.getRoomNumber() == room)
45         cout << "Passed";
46     else
47         cout << "Failed";
48 }
49
50
51 int main(void)
52 {
53     testSetterGetter(78.3, 343);
54     testSetterGetter(81.2, 326);
55     testSetterGetter(72.1, 101);
56
57     testConstructor(78.3, 343);
58     testConstructor(81.2, 326);
59     testConstructor(72.1, 101);
60
61     return(0);
62 }

```

```
1 # Makefile for tempReading class
2 # Complete except for one line
3
4 # (i) [5 pts] The line starting with CCC= is incomplete. Finish it.
5 # There are two possible answers.
6
7 CCC =
8
9 BINARIES = testTempReading
10
11 all: ${BINARIES}
12
13 # (j) [5 pts] The line below containing ${CCC} is incomplete; finish it!
14
15 testTempReading: testTempReading.o tempReading.o
16     ${CCC}
17
18 clean:
19     /bin/rm -f *.o a.out core ${BINARIES}
20
21
22
```

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;

    f = new double;
    g = &e;

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

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

11. (20 pts) Suppose you run the C++ program shown, and the program reaches the line of code that prints out Hi.

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

End of Exam. Total Points: 100 E

Key: version **E** CISC 181 sections 010-015, Midterm 2 04/19/06

1. (a)
2. (d)
3. (a)
4. (c)
5. (d)
6. (b)
7. (b)
8. (b)
9. (e)

Section 2. CISC181, E02, Spring 2006, Short Answer

```
10.1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #ifndef TEMPREADING_H
5 #define TEMPREADING_H
6
7 class TempReading_C
8 {
9 public:
10 TempReading_C(double theTemp, int theRoomNumber);
11
12 double getTemp() const;
13 int getRoomNumber() const;
14
15 void setTemp(double theTemp) ;
16 void setRoomNumber(double theRoomNumber) ;
17
18 private:
19 double temp;
20 int roomNumber;
21 };
22
23 #endif // TEMPREADING_H
```

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8 temp = theTemp;
9 roomNumber = theRoomNumber;
10 }
11
12 double TempReading_C::getTemp() const
13 {
14 return temp;
15 }
16
17 int TempReading_C::getRoomNumber() const
18 {
19 return roomNumber;
20 }
21
22 void TempReading_C::setTemp(double theTemp)
23 {
24 temp = theTemp;
25 }
26
27 void TempReading_C::setRoomNumber(double theRoomNumber)
28 {
29 roomNumber = theRoomNumber;
30 }
31 }
```

```
1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3
4 #include <iostream>
5 using std::cout;
6 using std::cerr;
```

```

7 using std::endl;
8
9 #include "tempReading.h"
10
11 void testSetterGetter(double temp, int room )
12 {
13
14     TempReading_C t(0.0, 0);
15
16     t.setTemp(temp); t.setRoomNumber(room);
17
18     if (t.getTemp() == temp && t.getRoomNumber() == room)
19         cout << "Passed";
20     else
21         cout << "Failed";
22
23 }
24
25
26
27 void testConstructor(double temp, int room )
28 {
29     TempReading_C x(temp, room);
30
31
32     if (x.getTemp() == temp && x.getRoomNumber() == room)
33         cout << "Passed";
34     else
35         cout << "Failed";
36
37 }
38
39
40 int main(void)
41 {
42
43     testSetterGetter(78.3, 343);
44     testSetterGetter(81.2, 326);
45     testSetterGetter(72.1, 101);
46
47     testConstructor(78.3, 343);
48     testConstructor(81.2, 326);
49     testConstructor(72.1, 101);
50
51     return(0);
52 }

```

```

1 # Makefile for tempReading class
2
3 CCC = CC
4
5 BINARIES = testTempReading
6
7 all: ${BINARIES}
8
9 testTempReading: testTempReading.o tempReading.o
10     ${CCC} testTempReading.o tempReading.o -o $@
11
12 clean:
13     /bin/rm -f *.o a.out core ${BINARIES}
14
15
16

```

11.

| <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 | error |
| &s | Point_S * |
| t->x | int |
| 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 * |

Name _____ F

UD Email address _____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

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!!!**

General Instructions

- The exam is 30% multiple choice, and 20% type expressions, and 50% “fill-in-the-blank” programming.
- The programming/short answer questions are on a separate sheet.
- Write your name **ONLY** in the spaces provided, so that the exam can be graded “name-blind”.
- You have 50 minutes. **Pace yourself**, and pay attention to the point values.
- Read *all* the directions *carefully* on each problem.
- Good luck.

1. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc
2 // P. Conrad for CISC181 exam
3
4 #include <iostream>
5 using namespace std;
6
7 void someFunction(const char *p)
8 {
9     char name[] = "fad";
10    cout << "p=" << p << endl;
11    p[0] = 'b';
12    p = name;
13 }
14
15 int main(void)
16 {
17     someFunction("foo");
18     return 0;
19 }
20
```

Which of the lines shown would produce a compiler error?

- (a) line 9
 - (b) line 10
 - (c) line 11
 - (d) line 12
2. (4 pts) C++ classes have a public part and a private part. This is most directly related to which aspect of Object-Oriented Programming?
- (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading

3. (3 pts) Consider the following program.

```
1 // testConstPointer01.cc
2 // P. Conrad for CISC181 exam
3
4 #include <iostream>
5 using namespace std;
6
7 void someFunction(char * const p)
8 {
9     char name[] = "fad";
10    cout << "p=" << p << endl;
11    p[0] = 'b';
12    p = name;
13 }
14
15 int main(void)
16 {
17     char president[6]="Adams";
18     someFunction(president);
19     return 0;
20 }
21
```

Which of the lines shown would produce a compiler error?

- (a) line 9
 - (b) line 10
 - (c) line 11
 - (d) line 12
4. (4 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

5. (4 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+0`
 - (c) `a++`
 - (d) `++a`
 - (e) `*a`
6. (4 pts) If the variable `p` is of type `double *`, which of the following expressions would “dereference” the variable `p`?
- (a) `deref(p)`
 - (b) `*p`
 - (c) `&p`
 - (d) `%p`
 - (e) `$p`
7. (2 pts) Which of the following terms is commonly used to refer to data members of a C++ class?
- (a) attributes
 - (b) parameters
 - (c) operands
 - (d) methods
8. (4 pts) Which of the following refers to the concept that C++ classes contain both data and functions in a single syntactic unit?
- (a) encapsulation
 - (b) information hiding
 - (c) inheritance
 - (d) polymorphism
 - (e) overloading
9. (2 pts) Which of the following terms is commonly used to refer to member functions of a C++ class?
- (a) attributes
 - (b) parameters
 - (c) operands
 - (d) methods

Section 2. CISC181, E02, Spring 2006, Short Answer

Name_____ F

UD Email address_____ @udel.edu

Please circle your section number:

010 (Mon 09:05) 013 (Wed 09:05)

011 (Mon 10:10) 014 (Wed 10:10)

012 (Mon 11:15) 015 (Wed 11:15)

Write your name and email ONLY in this spaces above.

It should appear nowhere else on this sheet of paper.

When finished, fold this page like a book with

- this as the “cover”
- question 10 on the inside
- question 11 on the “back”.

10. (50 pts) This sheet contains complete listings of four files, except that at a few places in the files, indicated by boxes and the labels (a) through (h), code has been removed.

Fill in the missing code.

Additional Information: These files **originally** contained complete correct source code for a C++ class specification, its member functions, a main program to test that class, and a Makefile to compile everything into an executable for the test program.

Each instance of this class represents a temperature reading taken in some room in Smith Hall. For example, an instance might represent the fact that the temperature in room 343 is 78.3 degrees.

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 // (a) [5 pts] Add something here to make this .h file idempotent
5
6
7
8 class TempReading_C
9 {
10 public:
11     TempReading_C(double theTemp, int theRoomNumber);
12
13     // (b) [5 pts] Fill in the function prototypes for the get functions
14
15
16
17
18     // rest of this file stays exactly as is
19
20     void setTemp(double theTemp) ;
21     void setRoomNumber(double theRoomNumber) ;
22
23 private:
24     double temp;
25     int roomNumber;
26 };
27
28 #endif // TEMPREADING_H
```

```

1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8     temp = theTemp;
9     roomNumber = theRoomNumber;
10 }
11
12 // (c) [10 pts] fill in the entire function definitions for the get functions
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27 void TempReading_C::setTemp(double theTemp)
28 {
29     // (d) [4 pts] In this functions, just fill in the body
30
31
32 }
33
34 void TempReading_C::setRoomNumber(double theRoomNumber)
35 {
36     // (e) [4 pts] In this function, just fill in the body
37
38
39
40 }

```

```

1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3 #include <iostream>
4 using std::cout;
5 using std::cerr;
6 using std::endl;
7
8 // (f) [4 pts] Something necessary is missing here
9 // Without it you will get the error
10 // Error: TempReading_C is not defined.
11
12
13
14 void testSetterGetter(double temp, int room )
15 {
16     TempReading_C t(0.0, 0);
17
18     // (g) [4 pts] Fill in code here to invoke the two setter methods
19     // on the object t, setting the temperature and room number
20     // from the values passed in to the function testSetterGetter
21
22
23     // Test whether the setter and getter methods are
24     // working correctly.
25
26     if (t.getTemp() == temp && t.getRoomNumber() == room)
27         cout << "Passed";
28     else
29         cout << "Failed";
30 }
31
32
33 void testConstructor(double temp, int room )
34 {
35
36     // (h) [4 pts] Fill in one line of code to invoke the constructor
37     // in such a way that the test below makes sense.
38
39
40
41     // Test whether the setter and getter methods are
42     // working correctly.
43
44     if (x.getTemp() == temp && x.getRoomNumber() == room)
45         cout << "Passed";
46     else
47         cout << "Failed";
48 }
49
50
51 int main(void)
52 {
53     testSetterGetter(78.3, 343);
54     testSetterGetter(81.2, 326);
55     testSetterGetter(72.1, 101);
56
57     testConstructor(78.3, 343);
58     testConstructor(81.2, 326);
59     testConstructor(72.1, 101);
60
61     return(0);
62 }

```

```
1 # Makefile for tempReading class
2 # Complete except for one line
3
4 # (i) [5 pts] The line starting with CCC= is incomplete. Finish it.
5 # There are two possible answers.
6
7 CCC =
8
9 BINARIES = testTempReading
10
11 all: ${BINARIES}
12
13 # (j) [5 pts] The line below containing ${CCC} is incomplete; finish it!
14
15 testTempReading: testTempReading.o tempReading.o
16     ${CCC}
17
18 clean:
19     /bin/rm -f *.o a.out core ${BINARIES}
20
21
22
```

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;

    e = new double;
    f = &d;

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

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

11. (20 pts) Suppose you run the C++ program shown, and the program reaches the line of code that prints out Hi.

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

End of Exam. Total Points: 100

Key: version **F** CISC 181 sections 010-015, Midterm 2 04/19/06

1. (c)
2. (b)
3. (d)
4. (b)
5. (e)
6. (b)
7. (a)
8. (a)
9. (d)

Section 2. CISC181, E02, Spring 2006, Short Answer

```
10.1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #ifndef TEMPREADING_H
5 #define TEMPREADING_H
6
7 class TempReading_C
8 {
9 public:
10 TempReading_C(double theTemp, int theRoomNumber);
11
12 double getTemp() const;
13 int getRoomNumber() const;
14
15 void setTemp(double theTemp) ;
16 void setRoomNumber(double theRoomNumber) ;
17
18 private:
19 double temp;
20 int roomNumber;
21 };
22
23 #endif // TEMPREADING_H
```

```
1 // tempReading.h temp readings in Smith
2 // P. Conrad for CISC181 Exam
3
4 #include "tempReading.h"
5
6 TempReading_C::TempReading_C(double theTemp, int theRoomNumber)
7 {
8 temp = theTemp;
9 roomNumber = theRoomNumber;
10 }
11
12 double TempReading_C::getTemp() const
13 {
14 return temp;
15 }
16
17 int TempReading_C::getRoomNumber() const
18 {
19 return roomNumber;
20 }
21
22 void TempReading_C::setTemp(double theTemp)
23 {
24 temp = theTemp;
25 }
26
27 void TempReading_C::setRoomNumber(double theRoomNumber)
28 {
29 roomNumber = theRoomNumber;
30 }
31 }
```

```
1 // testTempReading.cc
2 // P. Conrad CISC181 Exam E02
3
4 #include <iostream>
5 using std::cout;
6 using std::cerr;
```

```

7 using std::endl;
8
9 #include "tempReading.h"
10
11 void testSetterGetter(double temp, int room )
12 {
13
14     TempReading_C t(0.0, 0);
15
16     t.setTemp(temp); t.setRoomNumber(room);
17
18     if (t.getTemp() == temp && t.getRoomNumber() == room)
19         cout << "Passed";
20     else
21         cout << "Failed";
22
23 }
24
25
26
27 void testConstructor(double temp, int room )
28 {
29     TempReading_C x(temp, room);
30
31
32     if (x.getTemp() == temp && x.getRoomNumber() == room)
33         cout << "Passed";
34     else
35         cout << "Failed";
36
37 }
38
39
40 int main(void)
41 {
42
43     testSetterGetter(78.3, 343);
44     testSetterGetter(81.2, 326);
45     testSetterGetter(72.1, 101);
46
47     testConstructor(78.3, 343);
48     testConstructor(81.2, 326);
49     testConstructor(72.1, 101);
50
51     return(0);
52 }

```

```

1 # Makefile for tempReading class
2
3 CCC = CC
4
5 BINARIES = testTempReading
6
7 all: ${BINARIES}
8
9 testTempReading: testTempReading.o tempReading.o
10     ${CCC} testTempReading.o tempReading.o -o $@
11
12 clean:
13     /bin/rm -f *.o a.out core ${BINARIES}
14
15
16

```

11.

| <i>expression</i> | <i>type</i> |
|-------------------|-------------|
| a | int * |
| *a | int |
| &a | int ** |
| b | int |
| *b | error |
| &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 * |