

Name _____ Login name _____

General Instructions

- DO NOT PUT YOUR SSN ON ANYTHING!
- DO NOT WRITE YOUR NAME ON ANY PAGE EXCEPT THIS ONE!
- Turn off any noise making device, especially **CELL PHONES**. You may lose up to one letter grade if your device disturbs the peace of the exam.
- You have lots of minutes. Relax, **pace yourself**, and pay attention to the point values.
- The exam is points multiple choice, 100 pts programming and short answer.
- Do not add features that are not required by the problem. For example, if the instructions **don't say anything** about user input, then your program should **not** take user input. If you aren't sure, ask.
- Do problems you are confident about first. If you finish the problems you know, write what you do know about other problems to gain partial credit; but erroneous information may detract from that credit or irritate the grader, so don't make stuff up.
- Read *all* the directions *carefully* on each problem.
- Often writing a fast, rough version of a program in English or pseudocode will make your coding faster and more accurate. It also enables me to give partial credit in some circumstances.
- You may assume that input will not produce errors for the procedures described, unless the questions say otherwise.
- Do not do unnecessary testing. For example, testing for both $x < 0$ and $x \geq 0$ instead of using one test and then `else` would be considered unnecessary testing.
- Stuck? Confused? Need a reality check? Ask me.
- Have fun!

Error definition

Errors do not always create problems in output, but should be considered errors anyway, as we do in class. This means errors include, but are not limited to:

- incorrect number of parameters
- writing to invalid memory
- reading from invalid memory
- other compilation errors
- other runtime errors

Errors do not include type coercion in arithmetic (for example, assigning a double to an int) *unless* that impairs the correctness of a program.

6. (20 pts) Write a function that takes an array of char pointers and its size as parameters. The function reads words from user input, allocates space for each word, until the array is full.

7. (10 pts) Suppose you are **given** a function

```
void printStars(int n);
```

that prints a row of n asterisks. Using this function, write a **recursive** function triangle that prints a triangle of asterisks. For example, triangle(4) would print

```
****
***
**
*
```

8. (6 pts) Explain (you may use arrows or labels to annotate your previous code) how to minimally change the triangle function so that triangle(4) will now print

```
*
**
***
****
```

9. (10 pts) Given the following prototypes, write **only** the high level function "play" for the tictactoe assignment. Suggestion: think in terms of top-down design, as we did when we wrote it together in class, and sketch a pseudocode outline to the right before you try to write syntactically correct C++.

```
int countPlayerInColumn(char board[][SIZE], char player, int column);
double rateMove(char board[][SIZE], char player, int coords[]);
char other(char player);
void updateBoard(char board[][SIZE], char player, int coords[]);
bool isWin(char board[][SIZE], char player);
int countPlayerInDiagonal(char board[][SIZE], char player, int diag);
void getNextMove(char board[][SIZE], char player, int coords[]);
bool isDraw(char board[][SIZE]);
void draw(char board[][SIZE]);
int countPlayerInRow(char board[][SIZE], char player, int row);
int convertCoordRowToArrayRow(int coordRow);
```