CS 101, Spring 2016 — March 22nd — Exam 2 Name: _____

**Question 1**. [3 points] Which of the following loop statements would most likely cause the loop to execute *exactly* `n` times? You may assume that `n` will be set through user input.

(Circle one.)

a. `for(int i=0; n < i; n++)`

b. `for(int i=0; i < n; i--)`

c. `for(int i=1; i < n; i++)`

d. `for(int i=1; i <= n; i++)`

e. `while(n < 10)`

**Question 2**. [3 points] How many times will the following while loop execute? Provide a *brief* explanation for partial credit.

```
int i;
int num[10] = {1,2,3,4,5,6,7,8,9,10};
printf("Counting: ");
while (i < 10) {
   printf("%i ",num[i]);
}
```

(Circle one.)

a. 0

b. 1

c. 9

d. 10

e. infinite

f. cannot be determined

- i is uninitialized

**Question 3**. [3 points] Given the following symbolic constant declaraion

```
#define NUM_ELEMENTS 10
```

what is the proper way to declare an array named `cost` that can hold *at most* 11 `double` values?

(Circle one.)

a. int cost[NUM_ELEMENTS];

b. double cost[NUM_ELEMENTS]

c. double cost[NUM_ELEMENTS+1];

d. double cost[SIZE];

e. None of the above

**Question 4**. [3 points] Given the following array, what is the value of `quiz[4]`?

```
float quiz[5] = {88.9,  56.5, 92.1, 78.0, 68.9};
```

[0] [1] [2] [3] [4]

(Circle one.)

a. 0

b. 68.9

c. 78.0

d. 88.9

e. cannot be predicted

**Question 5**. [5 points] Identify and correct the error in the following code:

```
int arr[10] = {9,2,4,5,6,3,8,1,7,10};
int count = 0;
printf("Even #'s: ");
// find and print the even numbers in the array
for (int i = 0; i < 10; i++) {
  if (arr[i] % 2 == 0) {
     printf("%i ",arr);
     count++;
  }
}
printf("\n");
printf("There are %i even numbers.",count);
```

arr[i]

**Question 6**. [3 points] In the space provided below declare a two-dimensional array of decimal values called `grades` so that it has `NUM_STUDENTS` rows and `NUM_EXAMS` colums.

double grades[NUM_STUDENTS][NUM_EXAMS];

**Question 7**. [5 points] Consider the following code:

```
int value = (3 * (rand()%10)) - 1
printf("%i\n", value);
```

(a) What is the *smallest* possible value that can be printed by the `printf` statement?

-1

(b) What is the *largest* possible value that can be printed by the `printf` statement?

26

**Question 8**. [5 points] What output is printed by the following code?

```
for (int i = 2; i < 16; i *= 2) {
   printf("%i %i\n", i, (i+1)/2);
}
```

2 1
4 2
8 4

i
2
4
8
16

**Question 9**. [5 points] What output is printed by the following code?

```
for (int i = 0; i <=10; i++) {
   if (i%5 == 0) {
      printf("+");

   } else {
      printf("-");
   }
}
```

+ - - - - + - - - - +

i → (0) (1) (2) (3) (4) (5) (6) (7) (8) (9) (10)

**Question 10**. [5 points] What output is printed by the following code?

```
int count = 0;
for (int i = 0; i < 4; i++) {
   for (int j = 0; j < i; j++) {
      count++;
   }
   printf("Loop %i - %i\n",i,count);
}
printf("Final count: %i",count);
```

| i | j | count |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 2 | 1 | 2 |
| 3 | 0 | 3 |
| | 1 | |
| 4 | 2 | 4 |
| | 0 | |
| | 1 | |
| | 2 | |
| | 3 | |

Loop 0 - 0
Loop 1 - 1
Loop 2 - 3
Loop 3 - 6
Final count: 6

**Question 11**. [5 points] What output is printed by the following code?

```
int vals[6] = {0, -4, 7, 2, -2, 8};
for (int i = 0; i < 5; i++) {
   if (vals[i] < 0) {
      vals[i] = (-1)*vals[i];
   }
   printf("%i ",vals[i]);
}
```

0 4 7 2 2


**Question 12**. [5 points] What are the values in the array b[] after the following code executes?

Assume the symbolic constant `SIZE` has been defined as `#define SIZE 5`

```
int a[SIZE] = {2,4,6,8,10};
int b[SIZE] = {0,0,0,0,0};
for (int i = 0; i < SIZE-1; i++) {
   b[i] = a[i+1] - a[i];
}
```

(Circle one.)

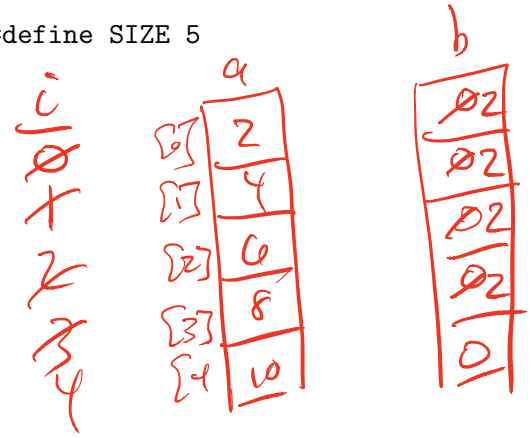
a. {0,0,0,0,0}

b. {2,2,2,2,2}

c. {2,2,2,2,0}

d. {-2,-2,-2,-2,-2}

e. cannot be determined because of out of bounds array access



# Programming Questions

**Note**: For all of the programming questions, you should use `scanf` to read the input value(s) required by the program.

**Note**: Make sure your programs produce the output in **exactly** the format described, including capitalization and punctuation. You may not receive credit for programs that produce incorrectly-formatted output.

**Getting started**: Start **Cygwin Terminal** and **Notepad++** and make sure ALL TABS are closed. (Note: do *not* open any other programs.) Your instructor will give you the name of a zip file. In your terminal, run the following commands:

```
cd h:
mkdir -p CS101
cd CS101
curl -O http://faculty.ycp.edu/~dbabcock/spring2016/cs101/zipfile
unzip zipfile
cd CS101_Exam2
```

Note that in the `curl` command, the `-O` has the letter 'O', not the digit '0'.

Substitute the name of the zip file for *zipfile*.

**Editing code**: Use your text editor to open the source file (e.g., `question13.cpp`) referred to in the question. Do not open any files other than the ones for the exam.

**Compiling**: To compile the program for Question 13, run the following command in the terminal:

```
make question13.exe
```

Change the number as appropriate for the other questions (e.g., `question14.exe`).

**Running**: To run the program for Question 13, run the following command in the terminal:

```
./question13.exe
```

Change the number as appropriate for the other questions (e.g., `question14.exe`).

**To submit**: In Cygwin Terminal, run the command

```
make submit
```

Enter your Marmoset username and password when prompted.

# Good luck!

**Question 13**. [15 points] Complete the program in `question13.cpp` so that it prints out the day numbers for all seven days in a week.

The program will read two integer input values. The first input value is the day number of Sunday, i.e., the first day of the week. The second input value is the number of days in the month, which is needed in case the week includes the last day of the current month and the first day of the next month. You may assume that the first value will always be less than or equal to the second value such that the starting day of the week is valid.

Example: if the input values are **10 28** (note that the end of the month *does not* occur), then the output should be

```
10 11 12 13 14 15 16
```

Example: if the input values are **28 31** (note that the end of the month *does* occur), then the output should be

```
28 29 30 31 1 2 3
```

In the second example, Thursday the 31st was followed by Friday the 1st, in a month with 31 days.

Hints:

- The program `does not` determine the particular month(s)
- Use a for loop that executes *exactly* 7 times
- Use a variable to keep track of the current day number: each execution of the loop should print and update this variable
- Print a space after each day number
- Print all of the day numbers on a single line of output
- Use an if statement to handle the case where the last day of the month changes to the first day of the next month

**Question 14**. [15 points] Complete the program in `question14.cpp` so that it first asks the user to enter the number of exam scores they wish to enter. The program should then prompt the user to enter that many *integer* exam scores (you may assume the values will be in the range 0-100 and **do not** need to check that they are valid).

The program should output the following values:

1. The **minimum** score entered
2. The **maximum** score entered
3. The **mean**, i.e. **average**, of the scores entered **AFTER** dropping the lowest score with 2 decimal places of precision.

Example run (user input in **bold**):

```
Enter number of exams: 4
Enter next exam score (0-100): 93
Enter next exam score (0-100): 87
Enter next exam score (0-100): 63
Enter next exam score (0-100): 82

Min: 63, Max: 93, Mean of 3 highest exams: 87.33
```

Another example run (user input in **bold**):

```
Enter number of exams: 3
Enter next exam score (0-100): 74
Enter next exam score (0-100): 86
Enter next exam score (0-100): 91

Min: 74, Max: 91, Mean of 2 highest exams: 88.50
```

Hints:

- This program can be written with or without using an array. If you choose to use an array, you may assume there will not be more than 10 scores entered.
- Initialize the minimum value to a value above the input range
- Initialize the maximum value to a value below the input range
- To drop the lowest exam score, remove the minimum from the total
- Remember to use the correct number of exam scores when finding the mean

**Question 15**. [20 points] Complete the program in `question15.cpp` as follows:

The program loads `num_vals` values into an integer array called `data` along with an integer target value `target`. This part of the program is provided: *do not change it*.

The program should then compare the sum of each triple, i.e. three consecutive values, in the array to the target value. It should count the number of and compute the total of all the triples that are

- Greater than the target
- Less than the target
- Equal to the target

Finally it should print out which group has the largest total.

Example run (user input in **bold**):

```
Enter number of values: 8
2 3 6 3 5 4 9 1
Enter the target value: 12
There are 3 triples greater than 12 with total 46
There are 1 triples less than 12 with total 11
There are 2 triples equal to 12 with total 24
Greater than has the largest total
```

The triples are $(2, 3, 6) = 11$, $(3, 6, 3) = 12$, $(6, 3, 5) = 14$, $(3, 5, 4) = 12$, $(5, 4, 9) = 18$, $(4, 9, 1) = 14$

Another example run (user input in **bold**):

```
Enter number of values: 6
2 2 2 2 1 1
Enter the target value: 6
There are 0 triples greater than 6 with total 0
There are 2 triples less than 6 with total 9
There are 2 triples equal to 6 with total 12
Equal to has the largest total
```

The triples are $(2, 2, 2) = 6$, $(2, 2, 2) = 6$, $(2, 2, 1) = 5$, $(2, 1, 1) = 4$

Hints:

- Use separate counters and accumulators for each group
- Be careful of array access out-of-bounds
- Make sure the four lines of output printed by the program use *exactly* the format shown in the examples