Programming 1 - Sample Milestone Assignment 2 Questions

Milestone 2 on 10/25/2021, in-class

Released: October 21, 2021

The Milestone Assignment consists of

1. 10 multiple choice questions - 30 points
2. 1 "find the output" question - 15 points
3. 2 code writing questions - 40 points
4. 1 code debugging question - 15 points
5. 3 short answer questions - 15 points

General details:

- **The Milestone Assignment will be on paper, in-class**
- The milestone assignment can be solved in about 50 minutes.
- The milestone assignment is supposed to be individual work. You will be asked to sign an Honor Code statement.
- The Milestone Assignment is closed-book, closed-notes. You are **NOT** allowed to use any electronic devices, including graphing calculators.
- You will have an opportunity to earn 15 extra credit points.
- The milestone assignment will examine your understanding of the material rather than your capacity to memorize copious amounts of information.
- Please try and attempt all questions. You get points for trying.
- Anything from the homeworks / quizzes / in class examples / exercises / slides is fair game. You don't need to look for more material.
- Code debugging is mostly syntax based (typos, missing brackets, semicolons, mismatched quotes, etc.)
- The code writing questions will be heavily based on the homeworks and class examples and exercises, with some modifications.
- The multiple choice and the debugging questions will test your familiarity with the C++ language and syntax. The code writing questions will test your knowledge of programming.
- **For Problem 2, you would need to explain you answer. Just writing out your output will only get you partial credit.**
- **For Problem 3, you will be graded based on**
  - Whether your code works

  - Are you using the required concepts?
  - Are you adhering to standard programming practices discussed in the course (no goto, auto, break outside a switch, continue; declaring and defining functions, etc).
  - IT IS MORE THAN JUST AN OUTPUT MATCHING PROBLEM
- **For Problem 4, you need to fix the error and explain what caused the error.** You will only be tested on syntax and semantic errors.
- **Solutions to the short answer questions must be SHORT and to the point. A 6 paragraph essay when 4 lines would do means your answer will just be skimmed through in the interest of grading time (The needs of the many ...)**
- You will also receive a Canvas announcement about the milestone assignment with these details on Saturday, 10/23/2021.
- **General Advice: For the sake of completing the assignment on time, please answer all questions before seeking to verify the answers**.
- Making me laugh might gain you points (depends on the quality of the joke).

Topics to study

- Basic C++ Syntax
  - Writing a basic C++ program stub: including required libraries, adding namespaces and writing the main function.
  - Simple statements - syntax.
  - Comments.
  - Reserved words, literals and escape sequences.
  - Style guidelines.
- Data types, variables, and sequential execution.
  - Naming, declaring and initializing variables.
  - Primitive data types.
  - Type Conversions - implicit and explicit.
  - Arithmetic Operators and operator precedence.
- I/O - printing and reading values from the user.
  - cout statements - printing literals and variables.
  - Precision for floating point variables.
  - Using cin to read data of different kinds.
- Selection statements and loops
  - Relational and logical operators.
  - Writing simple, multiple and nested if statements.
  - switch - case statements.
  - while, do-while and for loops
  - break and continue statements.
  - Overflow and Underflow

  - Scope of variable

- Writing functions in C++.
  - Writing simple functions.
  - Passing arguments and returning values. Pass by value and pass by reference.
  - Scope of local and global variables.
  - Function overloading and default parameters.

- Arrays
  - Declaring and initializing an array.
  - Reading in array values from the user and printing arrays.
  - Basic array operations - looking for a number, math with array elements, etc.
  - Passing arrays as parameters to functions.
  - Multi dimensional arrays.

- Strings
  - Cstrings - arrays of characters.
  - Reading strings with getline
  - cstring, cctype libraries
  - Basic string operations - counting different kinds of characters, finding and replacing substrings, etc.
  - Declaring, initializing, comparing and concatenating strings
  - Arrays of strings.

- Studying the topics listed above will be enough to pass the test. To get a 100, you would be required to study everything on the notes.
- You don't need to study from outside sources. The test is made entirely from the notes, class examples, exercises, quizzes and assignments.

Sample Questions

1. Which of the following is NOT a C++ reserved word?

   (a) float
   (b) function
   (c) void
   (d) return

   Answer: function, everything else is a keyword.

2. Given an array of doubles starting at address 5000, what is the starting address of element 7 of the array?

   (a) 5000
   (b) 5048
   (c) 5056

(d) 5064

Answer: 5000 + 7x8 = 5056

3. Which of the following functions will return 0 (false) for the character 'A'?

   (a) isalnum

   (b) isprint

   (c) islower

   (d) islapha

   Answer: islower, since 'A' is uppercase.

4. Write a C++ function that accepts a string as a parameter and replaces every occurrence of the letter 'e' with the character '3'. Then, in the main function, read in a string, and call the function with the input string as the parameter, and print the string. You can use C-Strings or C++ string objects. You can use the relevant libraries.
   Sample Run:
   ```
   Input:
   Never gonna give you up.  Never gonna let you down.  Never gonna run around and
   desert you.
   Output:
   N3v3r gonna giv3 you up.  N3v3r gonna l3t you down.  N3v3r gonna run around and
   d3s3rt you.
   ```

   Solution:

   ```
   #include <iostream>
   #include<cstring>
   using namespace std;

   void changeE(char str[]);
   int main() {
             char st[200];
           cout<<"Input:\n";
           cin.getline(st, 200);

           changeE(st);
           cout<<"Output:\n"<<str<<endl;
           return 0;
   }
   void changeE(char str[])
   {
       for(int i=0; i<strlen(st); i++)
       {
           if(str[i]=='e')
               str[i] = '3';
       }

   }
   ```

5. Write a C++ function that accepts a double array and its size. This array represents the side lengths of a bunch of squares. Calculate areas of squares, replace the array values with their squares, and return their sum. In the main function, create an array of 5 doubles. Read in

the values for the array, and then call the function. Finally, print the array and the returned sum. You do not have to account for floating point precision. You may use the `iostream` and the `cmath` libraries.
Sample Run:

```
Enter the array values:
12.5   9.7   4   2.63   7.1
The array is:
156.25  94.09   16  6.9169  50.41
The sum is 323.667
```

Solution:

```
#include <iostream>
using namespace std;

double sumSquares(double arr[], int size);

int main()
{
    double arr[5];
    cout<<"Enter the array values:\n";
    for(int i=0; i<5; i++)
    {
        cin>>arr[i];
    }

    double ans = sumSquares(arr,5);

    cout<<"The array is:\n";
    for(int i=0; i<5; i++)
    {
        cout<<arr[i]<<"\t";
    }
    cout<<"\nThe sum is "<<ans<<endl;

    return 0;
}

double sumSquares(double arr[], int size)
{
    double sum=0;
    for(int i=0; i<size; i++)
    {
        arr[i] = arr[i] * arr[i];
        sum = sum + arr[i];
    }

    return sum;
}
```

6. Figure out the output generated by the following code snippet:

```
int mat[3][3] = {{1,5,19},{6,-2,10},{12,8,5}};
int sum=0;
for(int i=0;i<3;i++)
        sum += mat[i][i];
cout<<"The sum is "<<sum<<endl;
```

Solution: The sum is 4

The code iterates through the rows of the matrix, and adds up the elements where the row number is equal to the column number. 1 + -2 + 5 = 4.

7. What is the difference between the size and the capacity of a string?
   Answer:
   The size of the string is the number of characters currently in the string. The capacity is the number of characters the string is *capable* of holding without going over bounds.

1 How to actually take the Milestone Assignment

This is a suggested method for tackling the milestone assignment and the key to completing it on time. Please note the emphasis on testing of your understanding of the material - programming is a process, so we're testing your understanding of the processes involved to accomplish the task using the tools you've learned, not your memorization skills.

1.1 Before the assignment

Practice is the only way to get better at programming. Please solve the practice problems. And then try to solve variations of the practice problems. For example, if the practice problem asked you to find the largest number, also try and find the smallest. If the practice problem asks you to run a loop forwards, also try running it backwards.

Similarly work through the class examples and homeworks. The problems on the test are heavily based on class examples, homework problems or practice problems.

If you have questions, ask during help sessions or after class. Help Sessions are not just for homework help. They're a vital resource when you don't have regular in-person contact with your instructor ot TA/LA.

1.2 During the assignment

TIME YOURSELF

Also, scratch paper and pencils are your best friends.

- 1 minute per multiple choice question - you either know it or can deduce it within a minute. Otherwise, just put down an answer and mark the question as a "review later" on your scratch paper.
- 10 minutes for the code reading question - work out the process on scratch paper. The "explanation" part can be short - explain the problem solved by the program - a line by line explanation is not required.
- 15 minutes per code writing problem. Turn in what you have. There is a lot of partial credit.

For this problem, you don't need to do error checking. You only need to solve the specific problem, assuming everything else goes well.

- 10 minutes for the code debugging question - we're only looking for syntax errors - errors that would stop CLion from compiling your program. Do not try to find the output for the program, or even see if the program makes sense line by line. Syntax errors are usually restricted to that line.

  You need to tell me what the error is and how it can be fixed - or show a line with the error fixed.

  We're aware errors propagate. So, if a future line will have an error due to previous line, and you've already identified and fixed the first, then this one wouldn't count. For example, if there is an error in a line with a variable declaration, then fixing that line would fix all the future "uninitialized variable" errors.

- 5 minutes per short answer question - Here, we're interested in what you know about the concept.

- The objective of the test is not to be 100% right. It mimics the programming interview, where a "quick and dirty" solution is good enough for the first part of the process - your initial thoughts - that demonstrates your level of knowledge and understanding of programming.