CSI 32
2 hours

Midterm Exam
SAMPLE
Answers

Part I

To be done without C++ compiler.
Answer all *True/False, Multiple choice*, and *short answer questions*

1. Which of the following is a compilation error?
(a) Neglecting to declare a local variable in a function before it is used.
(b) Using a triple equals sign instead of a double equals sign in the condition of an `if` statement.
(c) Omitting the left and right parentheses for the condition of an `if` statement.
(d) All of these.

2. What does a **.h** suffix at the end of a file name signify in C++?

The C++ header file is usually comes with an extension of .h

3. What is the difference between `=` and `==` ?

= is used for assignment statements
== is used for comparison, whether the value on the left is equal to the value on the right of ==

4. For each of the statements below state whether it is *true* or *false* and briefly explain why.

(a) The number and order of arguments in a function call must always match the number and order of parameters in the function definition's parameter list.

False, we can have default value parameters, and their values may be omitted during the function call.
Example: int f(int a, int b, int c=0);
the call f(4,5) is a legal call

(b) Every function body is delimited by an opening left square bracket and a closing right square bracket. Within the brackets are one or more statements that perform the function's task(s).

False, we should use curly braces, not square brackets.

5. What is string concatenation? Give an example:

string concatenation is when two strings are “merged” together.
For example, if str1 = “Hello” and str2 = “, how are you?” then the operation str1+str2 produces the string “Hello, how are you?”.

6. The return type _void__ indicates that when a function completes its task, it does not return (i.e., give back) any information to its calling function.

(a) `virtual`
(b) `null`
(c) `void`
(d) `nothing`

7. A function that should not modify the parameter passed by reference, should have _____ to the left of the parameter type.

(a) immutable
(b) firm
(c) const
(d) final

8. Having a loop within a loop is known as:

(a) Recursion.
(b) Doubling up.
(c) Stacking.
(d) Nesting.

Part II

To be done without C++ compiler. Answer all questions.

1. Consider the following code fragment. It has 3 syntax errors and a logical error.
Find all the errors and fix them.

```
// finds the sum of all positive integers up to n, i.e. 1 + 2 + … +n
int sum(int n); // prototype

int sum(int n) {  //return type int was missing

      if (n == 0) { return 0; } // semicolon was missing

      else {
            int result = 0;

            for(int c = 1; c > n; ++c)
            {   result += c;   }

            return result;
      }
} // the closing curly brace was missing
```

2. Consider the following code fragment.
What is the output of the program? What happens at each iteration of the for loop?

```
#include <iostream>

using std::cout;
using std::endl;

int f(int x, int y);

int main(){

      int x{1};

      for (int y = 20; y > 0; y -= 5)
      {
            cout << f(x,y) << endl;
            x++;
      }
}

int f(int x, int y) {
      return ((x - y) * (x + y));
}
```

Output:
-399
-221
-91
-9

The main function starts a loop with x = 1 and y = 20.
At every iteration of the loop:
1) the function f is called, with arguments x and
2) the function f returns a product of their difference and their sum, and the result is displayed, then
3) the value of x is incremented by 1, and
4) the value of y is decreased by 5.

3. Given the function below, add code that checks the size and throws an error if the size is a non-positive integer.

```
double average(double a[], int size) {
     // returns the average of values in the built-in array a,
     // if its size is positive

   if(size > 0) {

         double s=0;

         for (int i = 0; i < size; ++i) {

              s += a[i];
         }

         return s / static_cast<double>(size);
   }
   else
   {
      throw invalid_argument("error: the size of the array is not
positive.");
      or
      error("invalid array size");
      or
      throw -1; // not informational though

}
```

4. Recall *compile time functions*.What can we do to relay our intent of this function to perform its execution/calculation at the compile time?

1. The function must be declared as `constexpr`.
2. It should get constant expressions as argument
3. It cannot have any side effects, i.e. may not change the value of variables outside its own body
4. It should return a value
5. It may have a simple loop

Write a definition of a *compile time* function that returns the average of three decimal numbers.

```
constexpr double average(double a, double b, double c){
   return (a+b+c) / 3;
}


average(1,2,3); // ok, compile-time
cin >> x >> y >> z;
average(x,y,z); // not ok, not compile-time
```

5. For the program below, state the scope of each of the elements listed under it (*global scope, class scope, local scope* or *statement scope*) – see Section 8.4

```
#include <iostream>

using namespace std;

int f(int x, int y);  // function prototype, f is global

int y{20};

int main(){

     int x{1}; // x has local scope

     for (int y = 20; y > 0; y -= 5){

          cout << f(x,y) << endl; // y has statement scope,
                                  //operator<<  is global
          x++;
     }
}

int f(int x, int y) { // f is global
     return x * y;
}
```

(a) variable `x` in function `main`
(b) variable `y` in the `for` loop
(c) the function `f`
(d) `operator <<` of `cout`

Part III

Implement a guessing game, where the user is asked to guess a four-digit number. All digits of the number are different, unlimited number of attempts is given, and hints are given by the program (see below). The game stops when the user guessed the number.
The program should have a vector of four different digits, 0 – 9.
For example, say the number to be guessed is 1234.
The user guesses 1359; the program's response should be: 1 black and 1 red, because the user got one digit (1) right and in the right position (a black) and one digit (3) right but in the wrong position (red).
The game will continue until the user gets 4 blacks, that is all four digits are correct and are in correct order.

See the code posted on our website and Blackboard.