Summer 18 1020 Final Exam

Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Question 1: (3 points)

Consider the following program.  What is the output for this program.

#include<iostream>

using namespace std;

class Point

{

    private:

        int x, y;

    public:

        Point(){cout << "Default\n";}

        Point(int x1, int y1){   cout << "Param\n";}

        Point(const Point& p){ cout << "Copy\n"; }

        ~Point(){cout << "Destructor\n";}

        Point operator=(const Point& p){ cout << "operator=\n"; return \*this;}

};

int main()

{

      Point p1(10,15);

      Point p2(5,5)

      Point p3;

      p3 = p2 = p1;

      return 0;

OutPut:

}

Question 2: (3 points)

Given the following function prototypes, complete the code listed in main.

void printNum(int);

int adder(int, int);

int main(int argc, char\* argv[])

{

      1.Declare a function pointer called "fp" that can be pointed to printNum.

      2.Initialize the function pointer from 1 above to point to printNum.

      3.Call printNum using the function pointer passing the value of 54.

      return 0;

}

Question 3: (8 point). True / False

\_\_\_\_\_\_ Objects can be passed to functions, but they cannot be returned by functions.

\_\_\_\_\_\_When an object is passed to a function, but the function is not supposed to change it, it is best to pass the object as a reference.

\_\_\_\_\_\_ Suppose you have an abstract class called **Animal.** I can create a variable of type Animal by doing the following:

Animal\* aPtr;

\_\_\_\_\_\_ The following is an example of how a pure virtual function is declared?

class MyClass

{

      public:

      virtual myFunction() = 0;

};

Question 3 continued

\_\_\_\_\_\_\_ A reference variable can be initialized at any time, but a pointer must be initialized when created.

\_\_\_\_\_\_\_ Consider the linked list we discussed in class. If the self-referential pointer in a node has the value of NULL this indicates the end of the list.

\_\_\_\_\_\_\_ One advantage a vector has over a linked list is it is easier to insert an element in a vector than it is to insert a node in a linked list.

\_\_\_\_\_\_\_ Protected members are accessible to member functions of the base class as well as member functions of all derived classes.

Question 4(1 point)

What is the output of the following program?

OutPut:

#include <stdio.h>

void f(int\*, int );

int main()

{

    int i=5, j=10;

    f(&i, j);

    printf("%d\n", i+j);

    return 0;

}

void f(int\* p, int m)

{

    m = m + 5;

    \*p = \*p + m;

    return;

}

Question 5(2 points)

Type Compatibility:

Suppose that the classes **Dog** and **Ca**t derive from **Animal**.  Suppose further that **pDog, pCat**, and **pAnimal** are declared pointers to the respective classes.

The following code is incorrect and will cause an error.

pAnimal = new Dog;

pDog = pAnimal;

Rewrite the code so that if incorporated in a program it would not cause a compile error.

Question 6 (1 points)

With respect to inheritance:

When both a base class and a derived class have destructors. If an instance of the derived class goes out of scope the base class's destructor is called \_\_\_\_\_\_\_\_\_\_.

1. First
2. Last
3. Never

Question 7(1 point)

With respect to inheritance:

When both a base class and a derived class have constructors and an instance of the derived class is created the base class's constructor is called \_\_\_\_\_\_\_\_.

1. First
2. Last
3. Never

Question 8 (1 point)

Assume the following:

int var = 10;

Which of the following is a legitimate declaration of a C++ reference variable?

1. &int refvar;
2. int& refvar = var;
3. &int refvar = var;
4. &int refvar = 5;
5. int& refvar;

Question 9 (3 points)

Consider the following struct:

typedef Pixel{

   unsigned char red;

   unsigned char green;

   unsigned char blue;

};

In **C,** write the code to **dynamically allocate memory** for a **2D array** of **Pixels** of size **width** and **height.**  You can assume width and height have been declared and initialized.

I will not accept code that allocates for a 1D array, you must use the form we discuss in class.  (first allocate for the pointers, then allocate for the rows of cols)

Question 10 (3 points)

Consider the following struct:

typedef Pixel{

   unsigned char red;

   unsigned char green;

   unsigned char blue;

};

In **C++,** write the code to **dynamically allocate memory** for a **2D array** of **Pixels** of size **width** and **height.**  You can assume width and height have been declared and initialized.

I will not accept code that allocates for a 1D array, you must use the form we discuss in class.  (first allocate for the pointers, then allocate for the rows of cols)

Question 11 (2 points)

Briefly, describe what happens during compile time when you have an inline functions that is being called.

Question 12 (2 points)

In class we discussed two examples of how to make a function inline.  Using the following Class declaration provide the code -- for both examples -- to make the **add** function inline.

Class A{

      public:

          //this function returns a+b

          int add(int a, int b);

};

Question 13 (3 Points)

**Rewrite the following C program in C++**

#include <stdio.h>

int main(int argc, char\* argv[])

{

   FILE\* input = fopen(argv[1], "r");

   int num;

   while(fscanf(input, "%d", &num) == 1)

   {

      printf("%d\n" , num);

   }

   fclose(input)

   return 0;

}

Question 14 (2 points)

In class we discussed the "Rule of Three".  Briefly explain what this means.

Question 15 (9 points)

Consider the program labeled attachment A

Provide the implementation for the following three functions:

/\*Copy constructor\*/  
MyArray(const MyArray&);

/\*Overloaded operator+ \*/

friend MyArray operator+(const char, const MyArray&);

/\*A nice helper function would be a function that determines the length of the array. Below is the prototype of this function. Provide the implementation for this function.\*/

int MyArraylength(const char\* str);

Use the space below and the blank page after this page if you need additional space.

Question 16(4 points)

Consider the Time class provided on attachment B. Complete the two constructors as described below.

1. Implement the Default constructor using initialization list.
2. Implement the regular constructor.  **You are to insure the hour, minutes, and seconds are within the appropriate limits.** If they are not, print a statement indicating an invalid time has been entered and exit the program.  Do not use initialization list with this constructor.  **DO NOT USE MILITARY TIME.**

Question 17(8 points)

1. Base Class
2. Derived Class
3. Dynamically Allocated memory
4. Inheritance
5. Override of Function
6. Virtual Function
7. Fixed
8. Setprecision
9. Template
10. Polymorphic
11. Stack
12. Queue
13. Operator Overloading
14. Static member variable
15. Setw
16. Showpoint

\_\_\_\_Allows us to set the number of significant digits.

\_\_\_\_Describes a general class that other classes can inherit data members and functions, also known as a parent class.

\_\_\_\_ Initialized outside of the class declaration and is shared by all instances of the class.

\_\_\_\_Used by C++ to redefine how standard operators work when used with class objects.

\_\_\_\_Sets the size of a print field.

\_\_\_\_A data structure that stores and retrieves items in a first-in-first-out manner.

\_\_\_\_Code that produces different behavior when executing code with different types.

\_\_\_\_Forces a floating point number to display in a particular point format.

\_\_\_\_This is when a derived class provides functionality of a function with the same name, same parameters, and same return value of a function in the base class.

\_\_\_\_Describes a class that inherits functionality and data members from another class.

\_\_\_\_Allows us to define a class in terms of another class. Creates an “is-a” relationship.

\_\_\_\_A data structure that stores and retrieves items in a last-in-first-out manner.

\_\_\_\_A mechanism to create a generic function that can work with different data types.

\_\_\_\_Allows the most specific version of a member function in an inheritance hierarchy to be selected for execution. This is what makes polymorphism possible.

\_\_\_\_Causes a decimal point and trailing zeros to be displayed for floating-point numbers, evenif there is no fractional part.

\_\_\_\_This is when a program, while running, asks the computer to set aside a chunk of unused memory large enough to hold a variable of a specific data type.