**CS273 ASSIGNMENT #1: C++ Review**

## NAMES: Joseph Flinn DUE: Mon, Feb 9

## Received: \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_.

## GRADE:

|  |  |  |
| --- | --- | --- |
| **CATEGORY** | **POINTS** |  |
| **EX1\_01: Pointers** |  | **28** |
| **EX1\_02: Classes** |  | **24** |
| **EX1\_03: Templates/STL Vectors** |  | **18** |
| **EX1\_04: UML/Operator Overloading** |  | **40** |
| **TOTAL** |  | **110** |

**Objectives:**

* Review important concepts learned in CS172
* Prepare you for the task of implementing the advanced concepts and abstractions that we will be studying in CS273.
* Remind you that the best way to learn a programming concept is to implement a small test program that uses the concept.

**Instructions:**

**SAVE** your answers in this document and put it in the folder [\\CS1\CS\_STUDENT\your\_usename\CS273\HW1\](file:///\\CS1\CS_STUDENT\your_usename\CS273\HW1\) I also recommend you back up your assignments on a flash drive or other external backup.

**SUBMI**T a printed copy of this **cover sheet** either to the instructor OR to the TA on the due date.

**This may be one of the more important homework assignments**. It is important to try to complete this well, as everything we will be learning this semester will build on the concepts covered in this assignment.

* **Homework pre-requisite:**
  + **Please read Chapter P and sections 1.3 of your textbook.** We will not be going through this in class since you have all completed CS171 and CS172
  + **Please read appendix B.1 of your textbook to review how UML diagrams are drawn.**
  + It will also be very helpful to refer back to your CS172 textbook and homework assignments.
* **For most of the questions – please** **write a test program to verify that what you are writing actually works**. Points will be deducted from obvious syntax errors. Please allow Visual Studio to help you with this homework!

**EX1\_01: Pointers**

1. **2pts Declare** a pointer variable that can be used to store a **double** data type value.

double \*d;

1. **2pts** **Allocate memory** and assign “4.12” to the variable you just declared in the previous question.

double doub = 4.12;

d = &doub;

1. **2pts Return the memory** allocated in the previous question back to the operating system.

delete d;

1. **2pts What is wrong** with the following code?

char \* variable ;

variable = 3;

You can not assign a value to a pointer. You have to give it an address to where you want it to point.

1. **2pts What operator** do you use to get the address of the following variable, **myVar**?

**int myVar;**

One would use the & operator

1. **2pts** **Declare and allocate** an array of 10 **int** elements using **dynamic memory allocation**.

int \*nums = new int[10];

1. **4pts** **Write a C++ for loop** that assigns the value “42” to every element in the array allocated in the previous question **using pointer arithmetic to advance and access each element**.

for (int i = 0; i < ten; i++)

\*(nums+i) = 42;

1. **2pts Return the memory** allocated for the array in question 6.

delete[] nums;

1. **10pts Write the function** with the prototype

int \*doubleCapacity(int \*list, int size);

The function doubles the **size** (indicated by the parameter ***size***) of the array passed in through parameter ***list***.

(Hint: This was a CS172 homework assignment!)

int \*doubleCapacity(int \*list, int size)

{

// to store the initial values for later use

int \*oldList = list;

int newSize = size \* 2;

list = new int[newSize];

// adds the old data

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

list[i] = oldList[i];

delete[] oldList;

return list;

}

**EX1\_02: Classes**

1. **Declare a C++ class** called **Circle** that conforms to the following description:
   1. It has a **private** **property (member variable)** called **radius** that is of an **double** data type. (**2pts**)
   2. It has a **public** **default constructor** that initializes the **radius** of a circle object to “0” (**2pts**)
   3. It has a **public** **overloaded constructor** that takes a parameter to initialize the **radius** of a circle object (**2pts**)
   4. It has a **public method called getArea()** that returns the computed area (π\*radius\*radius) of the circle object. (**4pts**)
   5. It has **public** **getter** and **setter** methods for accessing and modifying the radius of a circle object respectively (**4pts**)

class Circle

{

private:

double radius;

public:

Circle(); // b

Circle(double); // c

double getArea(); // d

double getRad(); // e

void setRad(double); // f

};

#include "Circle.h"

// b

Circle::Circle() { setRad(0); }

// c

Circle::Circle(double rad) { setRad(rad); }

// d

double Circle::getArea() { return 3.14159265 \* getRad() \* getRad(); }

// f

double Circle::getRad() { return radius; }

// e

void Circle::setRad(double rad) { radius = rad; }

1. **2pts** **Declare** a **Circle** object variable, called **myCircle1,** using the default constructor.

Circle myCircle1;

1. **2pts Declare** a **Circle** object variable, called **myCircle2,** using the **overloaded constructor** to initialize its radius to “10”.

Circle myCircle2(10);

1. **2pts Declare a pointer variable** to a **Circle** object, and allocate memory for it using the overloaded constructor to initialize its radius to “12”.

Circle \*myCircle3 = new Circle(12.0);

1. **2pts Declare an array** of 10 **Circle** objects using the default constructor.

Circle circles[10];

1. **2pts** **Write a C++ for loop** that assigns the radius of every Circle object in the array defined in the previous question to “15”.

for (int i = 0; i < 10; i++)

circles[i].setRad(15);

**EX1\_03: Templates/STL Vector:**

1. **4pts** **Using templates, convert** the following Swap function to work with arguments of any generic type T:

void Swap(int & A, int & B) {

int tmp = A;

A = B;

B = tmp;

}

template <typename T>

void Swap(T &A, T &B)

{

T temp = A;

A = B;

B = temp;

}

1. **6pts** **Using templates, convert** the following class to hold an array of any generic type T:

class MyVec {

private:

int \*array; // dynamically allocated array

public:

MyVec(int size) { // constructor creates array of size “size”

array = new int[size];

}

~MyVec() { // destructor returns memory back to system

delete [] array;

}

};

template <typename T>

class MyVec

{

private:

T \*array;

public:

MyVec(int size) { array = new T[size]; }

~MyVec() { delete[] array; }

};

1. **2pts** **Declare an object of the template class** defined in the previous question (i.e. **MyVec**) to hold an array of **double** data types.

MyVec<double> mv(10);

1. **2pts** **Declare an STL vector** object for storing elements of the **char** data type.

vector<char> vec;

1. **2pts** **Write a C++ for** loop to assign a value (of your choice) to every element in the vector object declared in the previous question.

for (int i = 0; i < 10; i++)

vec.push\_back('a');

1. **2pts** **What STL vector method** do you use to get the current size of the vector?

I use the vector.size() method

## EX1\_04: UML/Operator Overloading

1. **4pt Draw a UML diagram** describing the composition relationship between the Elevator and Building class.

**class Elevator {**

Building

**…**

**};**

**class Building {**

**public:**

**Elevator elevator;**

**…**

Elevator

**};**

1. **4pt Draw a UML diagram** describing the inheritance relationship between the Square and Shape class.

**class Shape {**

Shape

**…**

**};**

**class Square : public Shape {**

**…**

**};**

Square

1. **1pt** When class A **inherits** class B, class A also inherits all public methods and variables in class B? (true/false)
2. **1pt** When class A **inherits** class B, can the **private** members variables of class B be accessed in class A? (yes/no)
3. **15pt Write C++ code for the classes “Animal”, “Cat”, and “Mouth”.** These classes must implement the following UML relationship, i.e. “Cat” is an “Animal”, and it has a “Mouth”.

The C++ classes do not need to have any other member variables or methods, other than what is needed to demonstrate the class relationship.

**Create a visual studio C++ project and define you classes in there.**

Your main() function should declare a **Cat** object.

**Save your project in the folder** [**\\CS1\CS\_Students\your\_username\CS372\HW1\inheritance**](file:///\\CS1\CS_Students\your_username\CS372\HW1\inheritance)**\**

Animal

Cat

Mouth

1. **15pts** Create a Visual Studio C++ project and **define the Circle class** used in EX1\_02. For the class, **define an operator + method** that will add 2 circles together to give you a new Circle with their radius added.

Your main() function should declare 2 circle objects of different radii. Add the 2 circles with “+” to derive a 3rd circle object with a combined radii.

**Save your project in the folder** [**\\CS1\CS\_Students\your\_username\CS273\HW1\operator\**](file:///\\CS1\CS_Students\your_username\CS273\HW1\operator\)

**(Operator overloading is a concept we will be using throughout this semester. Please review page 31 of your CS273 textbook, and chapter 14 of your CS172 textbook)**