**VINA LE**

CS271 – HW1

C++ Review

**EX1\_01: Pointers**

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

double \*pointVariable;

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

\*pointVariable = 4.12;

3. **2pts** Return the memory allocated in the previous question back to the operating system.

return pointVariable

4. **2pts** What is wrong with the following code?

char \*variable;

variable = 3;

variable = 3 is not stored. Would need \* for integer to be stored.

5. **2pts** What operator do you use to get the address of the following variable, myVar?

int myVar;

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

{

elements[i] = 42;

elements++;

}

return 0;

6. **2pts** Declare and allocate an array of 10 int elements using dynamic memory allocation.

delete myVar;

7. **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.

// the object has a function called size and returns the size

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

{ elements[i] = 42; }

Pointer Arithmetic:

for (int i=0;  I < 10; i++)

{ elements\*[i]=42 }

8. **2pts** Return the memory allocated for the array in question 6.

delete myVar;

9. **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(const int\* list, int size)

{

//double size

int \*list2 = new int[2 \* size];

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

{

list2[i] = list[i];

}

delete[] list; // removes old list

return \*list2;

}

**EX1\_02: Classes**

10. Declare a C++ class called Circle that conforms to the following description:

b.) It has a private property (member variable) called radius that is of an double data type. (**2pts**)

c.) It has a public default constructor that initializes the radius of a circle object to “0” (**2pts**)

d.) It has a public overloaded constructor that takes a parameter to initialize the radius of a circle object (**2pts**)

e.) It has a public method called getArea() that returns the computed area (π\*radius\*radius) of the circle object. (**4pts**)

f.) It has public getter and setter methods for accessing and modifying the radius of a circle object respectively (**4pts**)

11. **2pts** Declare a Circle object variable, called myCircle1, using the default constructor.

12. **2pts** Declare a Circle object variable, called myCircle2, using the overloaded constructor to initialize its radius to “10”.

13. **2pts** Declare a pointer variable to a Circle object, and allocate memory for it using the overloaded

constructor to initialize its radius to “12”.

14. **2pts** Declare an array of 10 Circle objects using the default constructor.

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

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

16. **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;

}

17. **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;

}

};

18. **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> (50);

19. **2pts** Declare an STL vector object for storing elements of the char data type.

vector <char> cVec[10];

20. **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++)

{

cVec[i].push\_back(‘t’);

}

21. **2pts** What STL vector method do you use to get the current size of the vector? cVec.size

**EX1\_04: UML/Operator Overloading**

22. **4pts** Draw a UML diagram describing the composition relationship between the Elevator and Building class.

class Elevator {

…

};

class Building {

Elevator

Building

public:

Elevator elevator;

…

};

23. **4pts** Draw a UML diagram describing the inheritance relationship between the Square and Shape class.

class Shape {

…

Shape

Square

};

class Square : public Shape {

…

};

24. **1pt** When class A inherits class B, class A also inherits all public methods and variables in class B? (true/false) TRUE

25. **1pt** When class A inherits class B, can the private members variables of class B be accessed in class A? (yes/no) YES

26. **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\<your\_student\_directory>\CS273-2\HW1\inheritance\](file:///\\CS1\%3cyour_student_directory%3e\CS273-2\HW1\inheritance\)

Animal

Cat

Mouth

27. 1**5pts** 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\<your\_student\_directory>\CS273-2\HW1\operator\](file:///\\CS1\%3cyour_student_directory%3e\CS273-2\HW1\operator\)