HANDOUTS | MATERIAL
HIGHLIGHTED AND PROVIDED BY :

VU TALK

CLICK TO JOIN OUR WHATSAPP GROUP :

VU TALK WHATSAPP GROUP

VU TALK YouTube CHANNEL : click me

For Any Help Contact Me On Whatsapp : click me

Lecture 34

1-Array of Code: The array of objects represent storing multiple objects in a single name. In an array of ob- jects, the data can be accessed randomly by using the index number. Reduce the time and memory by storing the data in a single variable.

New is called before the constructor, and that delete is called after the destructor. A global array called pool that can store all the Name objects expected.

2-Copying a File in the Reverse Order: Copying a File in the Reverse Order means the last byte of the input file will be the first byte of the output file, second last byte of the input file will be the second byte of the output file until the first byte of the input file becomes the last byte of the output file

The Reverse Order function outputs printed sheets in top-to-bottom order with the printed side facing up, resulting in a set of copies stacked in the opposite order to the original.

3-By overloading new and delete operators, only allocation and de-allocation part can be overridden.

4-By overloading the array operator ([]), one can implement mechanism to check for array bound.

5-With delete operator, destructor of the object is called first and then memory block is de-allocated.

6-With new operator function, a block of memory is allocated first and then constructor is called.

7-The delete operator returns nothing (void) and accepts a pointer of void to the memory block.

8-The new operator returns a void, accepts a parameter of type size_t.

9-The default constructor (parameter less constructor) is called for each element in the array allocated with new.

10 The default constructor is defined by the C++ compiler automatically for every class that has no default con- structor (parameter less constructor) defined already. 11- The same pointer that is returned by the new operator, is passed as an argument to the delete operator.

These rules apply to both, if operators (new and delete) are overloaded as member or non-member operators (as global operators).

Lecture 35

• A C++ class is a collection of data and the methods necessary to control and maintain that data.

• A class is a data type, analogous to ints, floats, and doubles. A C++ object is a specific varia- ble having a class as its data type, cin and cout are special pre-specified objects with different classes as their data types.

• A C++ stream is a flow of data into or out of a program, such as the data written to cout or read from cin.

• For this class we are currently interested in four different classes:

istream is a general purpose input stream. cin is an example of an istream.

●ostream is a general purpose output stream. cout and cerr are both examples of os- treams.

• ifstream is an input file stream. It is a special kind of an istream that reads in data from a data file.

●ofstream is an output file stream. It is a special kind of ostream that writes data out to a data file.

• Object Oriented Programming (e.g. C++) makes heavy use of a concept called inher- itance, in which some classes inherit the properties of previously written classes. The de- scendant classes then add on additional properties, making them specializations of their parent class.

For example, in the diagram below of (a portion of) the stream class hierarchy, we see that ifstream is a specialization of istream. What this means is that an ifstream IS an istream, and includes all the properties of the istream class, plus some additional properties of its own.

The ifstream Class

• An ifstream is an input file stream, i.e. a stream of data used for reading input from a file.

Because an ifstream IS an istream, anything you can do to an istream you can also do the same way to an ifstream.

• In particular, cin is an example of an istream, so anything that you can do with cin you can also do with any ifstream.

• The use of ifstreams (and ofstreams) requires the inclusion of the fstream header:

```
#include <fstream>
```

• Before you can use an ifstream, however, you must create a variable of type ifstream and connect it to a particular input file.

• This can be done in a single step, such as:

```
ifstream fin( "inputFile.txt");
```

• Or you can create the ifstream and open the file in separate steps:

ifstream fin; fin.open("inputFile.txt");

The ofstream Class

• An ofstream is an output file stream, and works exactly like ifstreams, except for output instead of input.

Once an ofstream is created, opened, and checked for no failures, you use it just like cout: ofstream fout "outputFile.txt");

fout << "The minimum oxygen percentage is" << min02 <<< endl;

Error Checking Using the Stream State

• Every stream has four bits that keep track of the current state of the stream:

The eofbit is set to true when an attempt is made to read past the end of the file.

• The badbit is set when corrupted data is read, i.e. when the type of data in the file does not match the type being read.

The failbit is set when a file fails to open, or when the end of file is read, or when cor- rupted data is read.

The goodbit is set to true whenever the other three bits are all false, and is false other- wise.

• The following methods are used to check and reset the bits:

●eof() returns the state of the eof bit.

bad() returns the state of the bad bit.

fail() returns the state of the fail bit.

good() returns the state of the good bit.

clear()-Sets the good bit to true and all others to false. This is needed to reset the state if asking the user to enter a new file name after a bad name was entered, or when re-us- ing a stream variable for a new file after encountering the end of a previous file.

Lecture 36

The manipulators are like something that can be inserted into stream, effecting a change in the behavior.

Non parameterized Manipulators do not take argument to control the formatting of input/output where as parameter- ized manipulators take argument for formatting.

Manipulators:

Manipulators are special functions that can be included in the I/O statement to alter the format parameters of a stream.

Manipulators are operators that are used to format the data display. •To access manipulators, the file iomanip, h should be included in the program.

Types of Manipulators in C++

They are of two types one taking arguments and the other without argument. Non-argument manipulators are also known as "Parameterized manipulators". These manipulators require lomanip header. Examples are setprecision, setw and setfill.

EXAMPLE OF MANIPULATORS:

During a disagreement or fight, a manipulative person will make dramatic statements that are meant to put you in a difficult spot. They'll target emotional weaknesses with inflammatory statements in order to elicit an apology. For ex- ample: "If you leave me, I don't deserve to live."

Lecture 37

In C++, stream insertion operator "<<" is used for output and extraction operator ">>" is used for in- put.

We must know the following things before we start overloading these operators.

1) cout is an object of ostream class and cin is an object of istream class

2) These operators must be overloaded as a global function. And if we want to allow them to access private data members of the class, we must make them friend.

The extraction operator (>>), which is preprogrammed for all standard C++ data types, is the easiest way to get bytes from an input stream object. Formatted text input extraction opera- tors depend on white space to separate incoming data values.

The insertion operator << is the one we usually use for output, as in: cout << "This is output" << endl; It gets its name from the idea of inserting data into the output stream.

If we overload insertion (<<) and extraction (>>) operators then the user of our class, does not need to know the specific names of the functions to input and display our objects.

Stream insertion (<<) and extraction operators (>>) are always implemented as non-member functions.

Operator << returns a value of type ostream & and operator >> returns a value of type istream & to support cascaded operations

Lecture 38

User Defined Manipulator:

C++ provides many predefined manipulators but you can also create yourown manipulators. The syntax for creating user defined manipulators is defined as follows:

```
ostream&manipulator_name(ostream&ostrObj)

set of statements;

return ostrObj;
```

Example:

```cpp
#include <iostream>
#include <iomanip>

ostream& curr(ostream& ostrObj) {
    cout << fixed << setprecision(2);
    cout << "Rs.";
    return ostrObj;
}

int main() {
    float amt = 10.5478;
    cout << curr << amt;

    return 0;
}
```

STATIC KEYWORD:

Static is a keyword in C and C++ which is used to declare a special type of a variable or a function inside or out- side of a class.

The static keyword is a non-access modifier used for methods and attributes. Static methods/attributes can be ac- cessed without creating an object of a class.

he static keyword can be used to declare variables and functions at global scope, namespace scope, and class scope. Static variables can also be declared at local scope. Static duration means that the object or variable is allo- cated when the program starts and is deallocated when the program ends.

STATIC OBJECT:

Static object is an object that persists from the time it's constructed until the end of the program. So, stack and heap objects are excluded. But global objects, objects at namespace scope, objects declared static inside classes/func tions, and objects declared at file scope are included in static objects.

Static data members of a class:

Static data members are class members that are declared using static keywords. A static member has certain spe cial characteristics. These are: Only one copy of that member is created for the entire class and is shared by all the ob- jects of that class, no matter how many objects are created.

Static data members of a class in namespace scope have external linkage. The initializer for a static data member is in the scope of the class declaring the member. A static data member can be of any type except for void or void quali- fied with const or volatile. You cannot declare a static data member as mutable.