NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

CL 103 - COMPUTER PROGRAMMING LAB

Instructors: Mr. Faizan Yousuf, Ms. Mahrukh Khan, Mr. Basit Ali, Ms. Rahemeen Khan
Email: faizan.yousuf@nu.edu.pk; mahrukh.khan@nu.edu.pk; basit.jasani@nu.edu.pk; rahemeen.khan@nu.edu.pk

Lab # 03

Outline

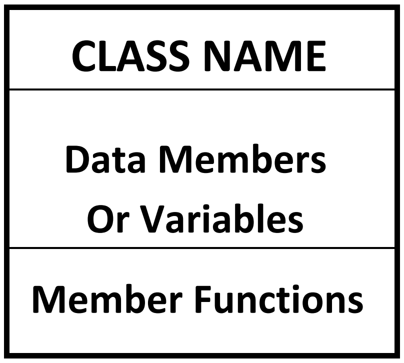
- Classes
- Objects
- Structures VS Classes
- Transformation from Procedural to Object Oriented Programming
- Example Programs
- Exercise

CLASSES

A class is a programmer-defined data type that describes what an object of the class will look like when it is created. It consists of a set of variables and a set of functions.

Classes are created using the keyword **class**. A class declaration defines a new type that links code and data. This new type is then used to declare objects of that class.

In the UML, a class icon can be subdivided into compartments. The top compartment is for the name of the class, the second is for the variables of the class, and the third is for the methods of the class.



```
class class-name
{
access-specifier:
data

access-specifier:
functions
};
```

CLASS NAME

By convention, the name of a user-defined class begins with a capital letter and, for readability, each subsequent word in the class name begins with a capital letter.

DATA MEMBERS

Consider the attributes of some real world objects:

RADIO – station setting, volume setting.
CAR – speedometer readings, amount of gas in its tank and what gear it is in.

These attributes form the data in our program. The values that these attributes take (the blue color of the petals, for example) form the state of the object.

MEMBER FUNCTIONS

Consider the operations of some real world objects:

RADIO – setting its station and volume (invoked by the person adjusting the radio's controls)
CAR – accelerating (invoked by the driver), decelerating, turning and shifting gears.
These operations form the functions in program. Member functions define the class's behaviors.

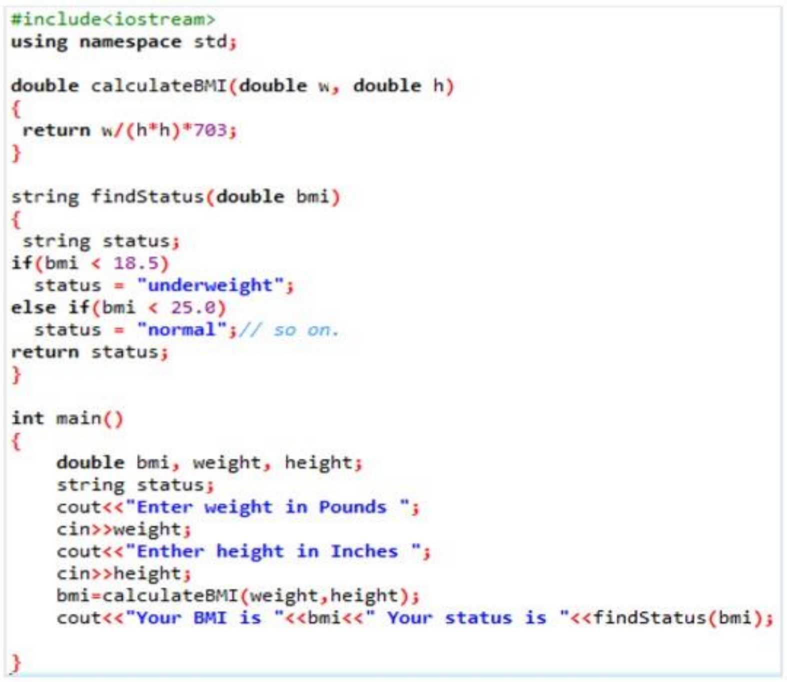
OBJECTS

In C++, when we define a variable of a class, we call it **instantiating** the class. The variable itself is called an **instance** of the class. A variable of a class type is also called an **object**. Instantiating a variable allocates memory for the object.
RADIO r;
CAR c;

Structures VS Classes

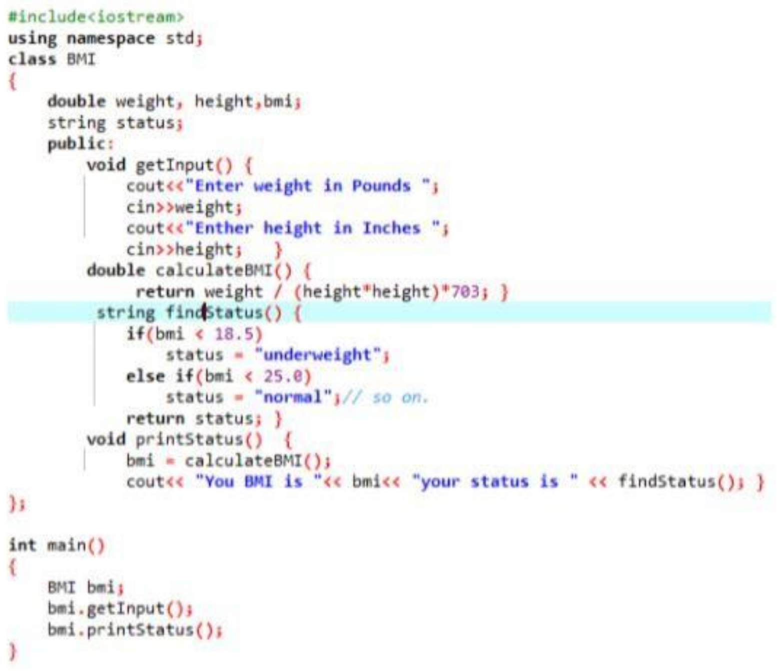
By default, all structure fields are public, or available to functions (like the main() function) that are outside the structure. Conversely, all class fields are private. That means they are not available for use outside the class. When you create a class, you can declare some fields to be private and some to be public. For example, in the real world, you might want your name to be public knowledge but your Social Security number, salary, or age to be private.

TRANSFORMATION FROM PROCEDURAL TO OBJECT ORIENTED PROGRAMMING



```
#include<iostream>
using namespace std;

double calculateBMI(double w, double h)
{
 return w/(h*h)*703;
}

string findStatus(double bmi)
{
 string status;
if(bmi < 18.5)
  status = "underweight";
else if(bmi < 25.0)
  status = "normal";// so on.
return status;
}

int main()
{
    double bmi, weight, height;
    string status;
    cout<<"Enter weight in Pounds ";
    cin>>weight;
    cout<<"Enther height in Inches ";
    cin>>height;
    bmi=calculateBMI(weight,height);
    cout<<"Your BMI is "<<bmi<<" Your status is "<<findStatus(bmi);

}
```

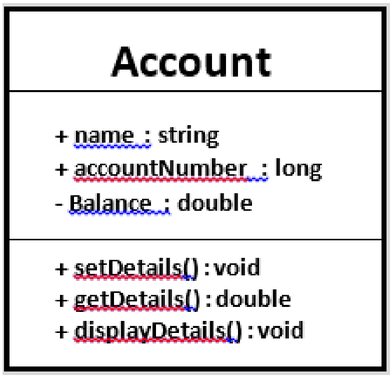
Procedural Approach



```
#include<iostream>
using namespace std;
class BMI
{
    double weight, height,bmi;
    string status;
    public:
        void getInput() {
            cout<<"Enter weight in Pounds ";
            cin>>weight;
            cout<<"Enther height in Inches ";
            cin>>height;    }
        double calculateBMI() {
             return weight / (height*height)*703; }
         string findStatus() {
            if(bmi < 18.5)
                status = "underweight";
            else if(bmi < 25.0)
                status = "normal";// so on.
            return status; }
        void printStatus()  {
            bmi = calculateBMI();
            cout<< "You BMI is "<< bmi<< "your status is " << findStatus(); }
};

int main()
{
    BMI bmi;
    bmi.getInput();
    bmi.printStatus();
}
```

Object Oriented Approach

EXAMPLE PROGRAM



```
#include<iostream>
using namespace std; class Account
{
private:
        double balance; // Account balance
public: //Public interface:
        string name; // Account holder
        long accountNumber; // Account number
        void setDetails(double bal)
        {
                balance = bal;
        }
        double getDetails()
        {
                return balance;
        }
```

Set and get functions to manipulate private data member

```
        void displayDetails()
        {
                cout<<"Details are: "<<endl;
           cout<<"Account Holder:
           "<<name<<endl;
                cout<<"Account Number: "<<
           accountNumber <<endl; cout<<"Account
           Balance: "<<getDetails()<<endl;
        }
};
int main(){
double accBal;
Account currentAccount;
currentAccount.getDetails();

cout<<"Please enter the details"<<endl;
cout<<"Enter Name:"<<endl; getline(cin,
currentAccount.name);
cout<<"Enter Account Number:"<<endl;
cin>>currentAccount.accountNumber;
```

Publically available data: Assigning values from

```
cout<<"Enter Account Balance:"<<endl;
cin>>accBal;
currentAccount.setDetails(accBal);
cout<<endl;
currentAccount.displayDetails();
return 0;
              }
```

Private data:
Accessing private data using member function

EXERCISE

Question # 1

Create a class of your name that has 4 member variables of types int, float, char and string. Take user input for these variables with a function named getdata() and display results with another function dispdata().

Question # 2

Write a program in which a class named student has member variables name, roll_no, semester and section. Create 4 objects of this class to store data of 4 different students, now display data of only those students who belong to section A.

QUESTION#3

Create a class Rectangle with attributes length and width, each of which defaults to 1. Provide member functions that calculate the perimeter and the area of the rectangle. Also, provide set and get functions for the length and width attributes. The set functions should verify that length and width are each floating-point numbers larger than 0.0 and less than 20.0.

QUESTION#4

Create a class called Employee that includes three pieces of information as data members—a first name (type char*), a last name (type string) and a monthly salary (type int). Your class should have a setter function that initializes the three data members. Provide a getter function for each data member. If the monthly salary is not positive, set it to 0. Write a test program that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10 percent raise and display each Employee's yearly salary again. Identify and add any other related functions to achieve the said goal.

QUESTION#5

Create a class called Invoice that a hardware store might use to represent an invoice for an item sold at the store. An Invoice should include four data members—a part number (type string), a part description (type string), a quantity of the item being purchased (type int) and a price per item (type float). Your class should have a functions that initializes the four data members. Provide a get function for each data member. In addition, provide a member function named getInvoiceAmount that calculates the invoice amount (i.e., multiplies the quantity by the price per item), then returns the amount as a float value. If the quantity is not positive, it should be set to 0. If the price per item is not positive, it should be set to 0. Write a test program that demonstrates class Invoice's capabilities.