NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

CL 217 – Object Oriented Programing

Lab 05

Outline

- this Pointer
- Constant Key word
- Static Key Word
- Exercise

What is "this" keyword?

- The current instance of the class is referred to by this keyword in C++ programming hence; this is known as this Pointer in C++.
- This pointer holds the address of current object, in simple words you can say that this pointer points to the current object of the class.
- Every object in C++ has access to its own address through an important pointer called this pointer. The this pointer is an implicit parameter to all member functions. Therefore, inside a member function, this may be used to refer to the invoking object.

The pointer acts as an implicit argument to all the member functions.

```
class ClassName {

      private:
        int dataMember;

      public:
         method(int a) {

  // this pointer stores the address of object obj and access dataMember

              this->dataMember = a;
             ... .. ...
          }
}
int main() {

   ClassName obj;
      obj.method(5);
   ... .. ...
 }
```

USE: When local variable's name is same as member's name To return reference to the calling object

1. **When local variable's name is same as member's name.**

Start here | LEC-05.CPP

```
1  #include <iostream>
2  using namespace std;
3  class Demo {
4  private:
5    int num;
6    char ch;
7  public:
8    void setMyValues(int num, char ch){
9      this->num =num;
10     this->ch=ch;
11   }
12   void displayMyValues(){
13     cout<<num<<endl;
14     cout<<ch;
15   }
16 };
17 int main(){
18   Demo obj;
19   obj.setMyValues(100, 'A');
20   obj.displayMyValues();
21   return 0;
22 }
23
```

- you can see that we have two data members num and ch. In member function setMyValues().
- we have two local variables having same name as data members' name.
- In such case if you want to assign the local variable value to the data members then you won't be able to do until unless you use this pointer,
- because the compiler won't know that you are referring to object's data members unless you use this pointer.

2. **When local variable's name is same as member's name (cont.)**

dsnslkgj.cpp

```
1  #include <iostream>
2  using namespace std;
3  #include <iostream>
4  using namespace std;
5  class example
6  {
7  private:
8  int x; public:
9  /* If function argument and data member is same then use this pointer
10 void set(int x)
11 {
12     this->x = x;
13 }
14 int get()
15 {
16 return x;
17 }
18 void printAddressAndValue()
19 {
20 cout<<"The adrress is "<<this<<" and the value is "<<x<<endl;
21 }
22 };
23 int main(){
24     example onj;
25     onj.set(5);
26     onj.printAddressAndValue();
27 }
```

```
The adrress is 0x22fe40 and the value is 5

--------------------------------
Process exited after 0.008007 seconds with return value 0
Press any key to continue . . .
```

3. **To return reference to the calling object**

```
#include <iostream>
using namespace std;
class Test{
        int x, y;
        public:
                Test(int x = 0, int y = 0){

                        this->x    =    x;

                        this->y = y;
                }
                Test& setX(int a){
                        x = a;
                        return *this;
                }
                Test& setY(int b){
                        y = b;
                        return *this;
                }
                void display(){

        cout<<"from"<<this<<"object
have x = "<<x<<"  y = "<<y<<endl;
                }
};
```

```
int main(){
        Test ob1;
        Test ob2;
        ob1.setX(10).setY(20);
        ob1.display ();//10 20        ob2.display(); // 0 0
        ob2 = ob1.setX (30);
        ob1.display ();    //20 30
        ob2.display ();    //20 30
        Return 0;
}
```

```
from0x22fe40object have x = 10  y = 20
from0x22fe30object have x = 0  y = 0
from0x22fe40object have x = 30  y = 20
from0x22fe30object have x = 30  y = 20

--------------------------------
Process exited after 0.01619 seconds with return value 0
Press any key to continue . . .
```

Defining Constants:

In C/C++ program we can define constants in two ways as shown below:

1. Using *#define* preprocessor directive
2. Using a *const* keyword

Constant Variables in C++

- If you make any variable as constant, using const keyword, you cannot change its value. Also, the constant variables must be initialized while they are declared.

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      const int b = 1;
6      b=4; //error: assignment of read-only variable 'b'
7      cout<<"here "<<b;
8     return 0;
9  }
```

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5     const int b; //error: uninitialized const 'b' [-fpermissive]
6     b=4;
7     cout<<"here "<<b;
8    return 0;
9  }
```

Constant Data Members In Classes

```
#include <iostream>
using namespace std;
class Students{
private:
string name;
const int rollno; // need to be initialized
once member is created
float cgpa;
```

```
public:
Students(int  rno):rollno(rno){} // using
memberinitializerlist
void set(string sname, float cg)
{ name = sname; cgpa = cg; }
void print()
```

```
{   cout<<"Name: "<<name<<", Roll #
"<<rollno<<",          CGPA          :
"<<cgpa<<endl;}};
int main ()
{
```

```
Students s(12);
s.set("Ahmad",3.67);
s.print();
}
```

Function with constant Arguments

- We can make the arguments of a function as const. Then we cannot change any of them.

```
void f(const int i)
{
 i++; // error
}
const int g()
{
        return 1; }
```

Defining Class Object as const

- When an object is declared or created using the const keyword,
- Its data members can never be changed, during the object's lifetime.

Syntax: const class_name object;

Defining Class's Member function as const

- A const member function never modifies data members in an object.
- Syntax:

return_type function_name () const;

Example for const Object and const Member function

```
class StarWars{
   public:
   int i;
   StarWars(int x)
   {  i = x;  }
int falcon() const {
cout << "Falcon has left the Base";
   }
int gamma()
   { i++;}
};
```

```
int main()
{
   StarWars objOne(10);       // non const object
   const StarWars objTwo(20);      // const object
   objOne.falcon();     // No error
   objTwo.falcon();     // No error
   cout << objOne.i << objTwo.i;
   objOne.gamma();     // No error
   objTwo.gamma();     // Compile time error
}
```

A static member is shared by all objects of the class

```
#include <iostream>
using namespace std;
class student
{
    public:
        int roo, age;
        static int m;

};
int student :: m=1;
int main (){

    student s1;
    s1.m = 5;
    student s2;
    cout <<s1.m<<endl<<s2.m<<endl;
    s2.m=10;
    cout <<s1.m<<endl<<s2.m<<endl;
    student s3;
    cout<<s3.m;
    return 1;
}
```

```
5
5
10
10
10
--------------------------------
Process exited after 0.01155 seconds with return value 1
Press any key to continue . . .
```

A static member function can only access static data member

```
1  #include <iostream>
2  using namespace std;
3
4  class Student {
5      public:
6          int rollnumber, age;
7          static int m;
8
9      static modifyAge () //Error
10     {
11         age = 10;
12     }
13 };
```

```
1  #include <iostream>
2  using namespace std;
3
4  class Student {
5      public:
6          int rollnumber, age;
7          static int m;
8
9      static modifyAge () //Not an Error
10     {
11         m = 10;
12     }
13 };
```

A static member function can only access other static member functions

```
1  #include <iostream>
2  using namespace std;
3
4  class Student {
5      public:
6          int rollnumber, age;
7          static int m;
8      void print()
9      {
10         cout<<"Hello"<<endl;
11     }
12     static modifyAge ()
13     {
14         m = 10;
15         print();  //Error
16     }
17 };
```

```
1  #include <iostream>
2  using namespace std;
3
4  class Student {
5      public:
6          int rollnumber, age;
7          static int m;
8      static void print()
9      {
10         cout<<"Hello"<<endl;
11     }
12     static modifyAge ()
13     {
14         m = 10;
15         print();  //Not an Error
16     }
17 };
```

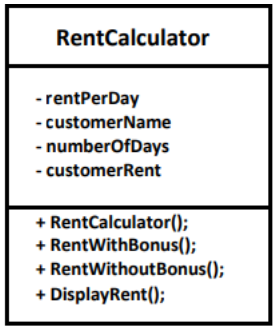
Activity

1. "Hotel Mercato" requires a system module that will help the hotel to calculate the rent of the customers. You are required to develop one module of the system according to the following requirements:
   a) The hotel wants such a system that should have the feature to change the implementation independently of the interface. This will help when dealing with changing requirements.
   b) The hotel charges each customer 1000.85/- per day. This amount is being decided by the hotel committee and cannot be changed fulfilling certain complex formalities.
   c) The module should take the customer's name and number of days, the customer has stayed in the hotel as arguments in the constructor. The customer name must be initialized only once when the constructor is called. Any further attempts to change the customer's name should fail.
   d) The module then analyses the number of days. If the customer has stayed for more than a week in the hotel , he gets discount on the rent. Otherwise, he is being charged normally.
   e) The discounted rent is being calculated after subtracting one day from the total number of days.
   f) In the end, the module displays the following details:

a. Customer Name
b. Days
c. Rent and discount

Note that, the function used for displaying purpose must not have the ability to modify any data member.

The following class structure must be followed.



- Use appropriate data types, return types and function arguments.
- Display the results for two initialized instances

REQUIRED OUTPUT

```
CustomerName: Dummy1
Days: 7
Rent: 7005.95
CustomerName: Dummy2
Days: 8
Rent: 7005.95

--------------------------------
Process exited after 0.06338 seconds with return value 0
Press any key to continue . . .
```

2. Create a class 'Employee' having two data members 'Employee Name' (char*) and 'EmployeeId' (int).Keep both data members private. Create three initialized objects 'Employee1', 'Employee2' and 'Employee3' of type 'Employee' in such a way that the employee name for each employee can be changed when required but the employee Id for each employee must be initialized only once and should remain same always. Use member initializer list, accessors and mutators for appropriate data members. The result must be displayed by calling the accessors. All of the accessors must not have the ability to modify the data.

3. Define a class to represent a Bank account. Include the following members.

a. Data members: -
b. Name of the depositor
c. Account number.
d. Type of account.
e. Balance amount in the account.
f. Rate of interest

Provide a default constructor, a parameterized constructor to this class.

Also provide Member Functions: -

1. To depositamount.
2. To withdraw amount after checking for minimumbalance.
3. To display all the details of an accountholder.
4. Display rate of interest (a static function)

Illustrate all the constructors as well as all the methods by defining objects.