ITEC243 – Lecture Session – 13-APR-2020

More examples on Dynamic Memory Management

Example 1:

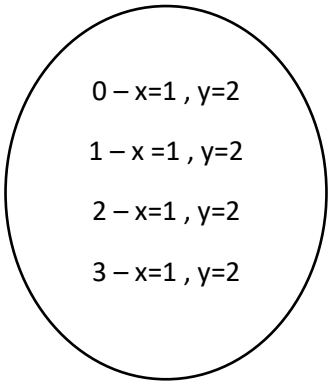
```
//point.h
class point{
	int x, y;
public:
	point()
	{
		this->x = 1;
		this->y = 2;
	}
	void print()
	{
		cout << "(" << this->x << "," << this->y << ")" << endl;
	}
	void set(int x, int y)
	{
		this->x = x; // x=x;???
		this->y = y;
	}
	~point()
	{
		cout << "Object with values:" << this->x << " " << this->y << " has been
deleted." << endl;
	}
};
```

//point.cpp

```
#include<iostream>
using namespace std;
#include"point.h"
void main()
{
	//create dynamic object- dynamic objects are created along with pointers!!!
	point *a;
	a = new point[4]; //initializing the pointer object here!!!
	//and default constructor is executed automatically
	cout << a << endl; //now the pointer object has a value.
	for (int i = 0; i < 4; i++)
		a[i].print();
	delete[]a; //destructor is exectued automatically
	system("pause");
}
```

Output:

```
014E4964
(1,2)
(1,2)
(1,2)
(1,2)
Object with values:1 2 has been deleted.
Object with values:1 2 has been deleted.
Object with values:1 2 has been deleted.
Object with values:1 2 has been deleted.
Press any key to continue . . .
```



Example 2: (second main function for the same header (point.h)

```
//point2.cpp
#include<iostream>
using namespace std;
#include"point.h"
void main()
{
point *a[4];// array of pointers object
a[0] = new point;
a[0]->set(4, 5);
delete a[0]; //delete[]a;//deletes the whole array object
for (int i = 0; i < 4; i++)
{
 a[i] = new point;
 a[i]->set(i, i + 1);
 a[i]->print();
}
delete a[1];
system("pause");
}
```

0 – X= ~~1~~ 4 ,Y= ~~2~~ 5
1
2
3

After the line -- delete a[0];

0 – x=~~1~~ 0 , y=~~2~~ 1	i=0
1- X=~~1~~ 1 , y=~~2~~ 2	i=1
	i=2
2- X=~~1~~ 2 , y=~~2~~ 3	i=3
3- X=~~1~~ 3 , y=~~2~~ 4	

Output:
(0,1)
~~(1,2)~~
(2,3)
(3,4)

FRIEND FUNCTIONS

A friend function is a special function which is **not a member of a class,** but **has direct access** to the **private** and **protected** members of the class. (Data-hiding (private data members) principle is very important in C++).
A friend function of a class is defined **outside that class's scope**.

- If a friend function (FF) is to be made friend of a class then its ***prototype*** has to be declared within the body of the class preceded with the keyword ***friend***.
- FF are neither public nor private, and **it can be declared anywhere inside the class**.
- Whenever a FF is defined, neither the **name of the class** nor **scope resolution operator** appears in its definition (normally we have to use name of the class as well as the (: :) operator as void rectangle::setLength(int length))
- Whenever a FF is called, neither the name of the object nor dot operator appear.
- If a FF wants to **manipulate** the values of the data members of an object, it needs to **reference (&)** the object to be passed as parameter.
- Friends are not symmetric. That is, If Class1 is a friend of Class2, it does not imply that Class2 is a friend of Class1!!!!
- Friends are also not transitive. That is, if Class1 is a friend of Class2, and Class2 is a friend of Class3, it does not imply that Class1 is a friend of Class3.
- Using FFs enhances the performance of the code.
- Use friend feature with CARE!!! Incorrect use of friends may corrupt the concept of information hiding and encapsulation principle.

In **composition** (employees and departments classes. Assume we have link (relationship) between employees and departments, e.g. employees work for a department). We create an object of a class inside another class!!!!
In **Inheritance** (employee class has types, e.g. Hourly_Paid, Regular_EMPs). To show is-a relationship between the class (super-class and its sub-classes) we use inheritance feature

of C++. In Relational DBs we cannot implement inheritance relationship between the entities/tables. But in C++, we can implement inheritance relationship between the classes in OOP!!!!

Example:

```
//example.h
class example{
private:
        int num;
public:
        example()
        {
                this->num = 0;
        }
        friend int getnum(example); //prototype of FF
        friend void setnum(example&, int);
};
//definition of FF has to be outside the class
int getnum(example eobj)
{
        return eobj.num;
}
void setnum(example& eobj, int a)
{
        eobj.num = a;
}

//example.cpp
#include<iostream>
using namespace std;
#include"ex1_ff.h"
void main()
{
  example e;
  cout << "Friend function returns the value of the private data member of the object ="
        << getnum(e) << endl;
  setnum(e, 5);
  cout << "After calling setnum() function we manipulated the private data
          member of the class ="<< getnum(e) << endl;
        system("pause");
}
```