	AIR UNIVERSITY
	DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
	EXPERIMENT NO 10

Lab Title: Dynamic Memory Allocation

Student Name: ______________________ **Reg. No:** ______________

Objective: ____

____

LAB ASSESSMENT:

Attributes	Excellent (5)	Good (4)	Average (3)	Satisfactory (2)	Unsatisfactory (1)
Ability to Conduct Experiment					
Ability to assimilate the results					
Effective use of lab equipment and follows the lab safety rules					

Total Marks: ______________ Obtained Marks: ______________

LAB REPORT ASSESSMENT:

Attributes	Excellent (5)	Good (4)	Average (3)	Satisfactory (2)	Unsatisfactory (1)
Data presentation					
Experimental results					
Conclusion					

Total Marks: ______________ Obtained Marks: ______________

Date: ______________ Signature: ______________

EXPERIMENT NO 10

Dynamic Memory Allocation

Objectives

- To understand and implement the core concept of dynamic memory in OOP classes

Equipment required

- Visual Studio/Dev C++

Description

A variable that is created during program execution is called **dynamic variable**. The dynamic variables can be created in C++ using pointers.

C++ provides two operators new and delete operators for dynamic variables.

The new operator is used to create dynamic variables and delete operator is used to delete dynamic variables during program execution.

The new operator

The new operator is used to allocate memory dynamically.

It is followed by the type of object for which the memory is to be allocated. The compiler allocates the amount of memory according to the type of object**. The new operator returns a memory address**. The returned address must be assigned to a pointer. The pointer is then used to access the memory location and process the values stored in that address.

The new operator can be used to create simple variable, an object or an array of objects.

The delete operator

The delete operator deallocates the memory and returns the allocated memory back to the free store. It takes a pointer as parameter and release the memory referred by the pointer. The dynamically created object should be deleted when it is not more required.

The pointer declared in a function is treated as a local variable. The pointer goes out of scope and is lost when the control exists from the function in which the operator is declared. However, the memory allocated with new operator is not freed automatically. That memory becomes unavailable. This situation is known as **memory leak**. This memory cannot be recovered until the program ends.

Pointers

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

int main(){
	int a = 5, *ptr;
	ptr = &a;
	cout << "Address of a = " << ptr << endl;
	cout << "Value of a = " << *ptr << endl;
	return 0;
}
```

Output

```
Address of a = 0x6ffe04
Value of a = 5
```

Pointers to Objects

A pointer can also refer to an object of class. The member of an object can be accessed through pointers using the symbol ->. The symbol is known as **member access operator**.

Syntax

The syntax of referencing an object with pointer is as follows:

ptr -> member

ptr (it is the name of pointer that references an object).

-> (it is the member access operator that is used to access a member of object).

member (it is the name of the class member to be accessed).

Example

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

class A{
	int a;
	public:
		void input(){
			cout << "Enter value of a = ";
			cin >> a; }
		void display(){
			cout << "The value of a is = " << a << endl; }
};

int main(){
	A *ptr, p;
	ptr = &p;
	ptr->input();
	ptr->display();
	return 0; }
```

Output

```
Enter value of a = 1
The value of a is = 1
```

Allocating memory dynamically

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

class A{
	int a;
	public:
		void input(){
			cout << "Enter value of a = ";
			cin >> a;
		}
		void display(){
			cout << "The value of a is = " << a << endl << endl;
		}
};

int main(){
	A *ptr;
	ptr = new A;	// creating object of class A (dynamically allocating memory)
	ptr->input();
	ptr->display();
	delete ptr;	// deallocating memory
	return 0;
}
```

Output

```
Enter value of a = 1
The value of a is = 1
```

Array of pointers to objects

An array can store the same type of data. An array of pointers can store memory addresses of the objects of same class. It allows us to create a large number of objects in memory using new operator and process them easily using loops. Each element of the array will refer to a different object in memory.

Example

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

class A{
	int a;
	public:
		void input(){
			cout << "Enter value of a = ";
			cin >> a; }
		void display(){
```

```
                              cout << "The value of a is = " << a << endl << endl; }
};

int main(){
        A *ptr[3];   // Array of pointers

        for(int i=0; i<3; i++){
                ptr[i] = new A;      // creating object of class A
                ptr[i]->input();
                ptr[i]->display();
        }

        for(int i=0; i<3; i++){
                delete ptr[i];
        }

        return 0;
}
```

Output

```
Enter value of a = 1
The value of a is = 1

Enter value of a = 2
The value of a is = 2

Enter value of a = 3
The value of a is = 3
```

Lab Tasks

1. Write a class that contains the following data member.
    - an integer

   The class contains the following member functions.

    - a function to input
    - a function to display it

   Create an object of class using pointer and calls its member functions.

2. Write a class that contains the following data members.
    - Name
    - Age
    - Gender

The class contains the following member functions.

- a function to input the data members
- a function to display the data members.

Create array of pointers in which each element refers to an object of the class.

3. Write a class that contains the following data member.
    - Marks (an array)
    - Number of subjects

   The class contains the following member functions.
   - It inputs the marks of students.
     - ✓ The function inputs the number of subjects from user.
     - ✓ It then declares a dynamic array of marks with size equal to number of subjects.
     - ✓ Takes marks of each subject from the user.
   - displays the average marks of the whole class.

   The class inputs the number of students in a class from user.
   It then declares a dynamic array of pointers with same number of elements.

4. Write a program that declares a class to store the
    - account id
    - amount.

   It inputs the number of account holders from the user and creates a dynamic array of classes to store the record of accounts.
   The program should declare two functions i.e., one for getting input from the user and the other for showing records to the user.