# UEE1302(1066) F12: Introduction to Computers and Programming

## Lab 12: Pointer & Dynamic Array (II)

### *What you will learn from Lab 12*

In this laboratory, you will understand how to use pointers to structure and manipulate dynamic arrays.

### TASK 12-1 : FUNCTION PASSING DYNAMIC ARRAY

✓ Program `lab12-1`, a revised version of program `dynamic array`, introduces how to use pointers as functional arguments and how to use return by a pointer.

```
//File: lab12-1.cpp
#include <iostream>
using namespace std;

int *New1DArray(const int);
void Initial1DArray(int *,const int);
void Display1DArray(int *,const int);
void Delete1DArray(int *);

int main()
{
    int n;
    cout << "Enter the size of array: ";
    cin >> n;

    int *vec = New1DArray(n);
    Initial1DArray(vec,n);
    Display1DArray(vec,n);
    Delete1DArray(vec);

    return 0;
}

int *New1DArray(const int n)
{
    int *vec = new int[n];
    return vec;
}
void Initial1DArray(int *vec, const int n)
{
    for (int idx = 0;idx < n;idx++)
        vec[idx] = idx;
}
```

```
void Display1DArray(int *vec, const int n)
{
   for (int idx = 0;idx < n;idx++)
      cout << vec[idx] << " ";
   cout << endl;
}
void Delete1DArray(int *vec)
{
   delete []vec;
}
```

✓ Program lab12-2 shows an example of the use of two-dimensional dynamic arrays.

```
//File: lab12-2.cpp
#include <iostream>
using namespace std;

int **New2DArray(const int,const int);
void Initial2DArray(int **,const int,const int);
void Display2DArray(int **,const int,const int);
void Delete2DArray(int **,const int);

int main()
{
   int m,n;
   cout << "Enter the size of column: " << endl;
   cin >> m;
   cout << "Enter the size of column: " << endl;
   cin >> n;

   int **vec = New2DArray(m,n);
   Initial2DArray(vec,m,n);
   Display2DArray(vec,m,n);
   Delete2DArray(vec,n);

   return 0;
}

int **New2DArray(const int m, const int n)
{
   int **mat = new int *[m];
   for (int idx = 0;idx < m; idx++)
      mat[idx] = new int [n];
   return mat;
}
void Initial2DArray(int **mat, const int m, const int n)
{
   for (int idx = 0;idx < m;idx++)
   {
      for (int jdx = 0; jdx < n; jdx++)
         mat[idx][jdx] = idx + jdx;
```

```
    }
}
void Display2DArray(int **mat, const int m, const int n)
{
    for (int idx = 0;idx < m; idx++)
    {
        for (int jdx = 0;jdx < n; jdx++)
            cout << mat[idx][jdx] << " ";
        cout << endl;
    }
}
void Delete2DArray(int **mat, const int n)
{
    for (int idx = 0; idx < n; idx++)
        delete [] mat[idx];
    delete []mat;
}
```

## TASK 12-2 : POINTER TO STRUCTURE

✓ Program `lab12-3` is an example of using the specific function to construct the elements in a structure.

```
// file: lab12-3.cpp
#include <iostream>
#include <string>
using namespace std;

typedef struct{
    long id;
    string name;
    int age;
    char gender;
} student;

void DisplayStudent(const student &);
student AssignStudent(long, string, int, char);

int main()
{
    student uee_a = AssignStudent(9823014,"Tommy",20,'F');
    student uee_b = AssignStudent(9823183,"Marry",21,'M');

    DisplayStudent(uee_a);
    DisplayStudent(uee_b);

    return 0;
}

student AssignStudent(long id, string name, int age, char gender)
```

```
{
    student member;
    member.id = id;
    member.name = name;
    member.age = age;
    member.gender = gender;
    return member;
}

void DisplayStudent(const student &member)
{
    cout << "name: " << member.name << endl;
    cout << "id: " << member.id << endl;
    cout << "age: " << member.age << endl;
    cout << "gender: " << member.gender << endl;
}
```

- The assignment of the structure is done in function `AssignStudent` to avoid repeating syntax in one program.

✓ Program `lab12-4` is a modified version of program `lab12-3`. In the program, the object is constructed by a pointer and the new operator. The return type of assignment function is modified as return by pointer. However, the argument list in display function is the same. You should observe how to pass a pointer to a function.

```
// file: lab12-4.cpp
#include <iostream>
#include <string>
using namespace std;

typedef struct{
    long id;
    string name;
    int age;
    char gender;
} student;

void DisplayStudent(const student &);
student *AssignStudent(long, string,int,char);

int main()
{
    student *uee_a = AssignStudent(9823014,"Tommy",20,'F');
    student *uee_b = AssignStudent(9823183,"Marry",21,'M');

    DisplayStudent(*uee_a);
    DisplayStudent(*uee_b);

    delete uee_a;
    delete uee_b;
```

```
    return 0;
}

student *AssignStudent(long id, string name, int age, char gender)
{
    student *member = new student;
    member->id = id;
    member->name = name;
    member->age = age;
    member->gender = gender;
    return member;
}
void DisplayStudent(const student &member)
{
    cout << "name: " << member.name << endl;
    cout << "id: " << member.id << endl;
    cout << "age: " << member.age << endl;
    cout << "gender: " << member.gender << endl;
}
```

- Note that you have to use operator `->` to access the member of a pointer type object.

## TASK 12-3 : EXERCISES

✓ Please write a program to explore the medium number(s) from user's input sequence. You have to implement this function by dynamic array.

Dynamic array provides programmer to use memory space more flexibly. However, you still need to assign an initial length to the dynamic array. Please implement the function ***resize***() to relax the space of the array by doubling the original size. For manipulating the memory space safely, you should test the size before you use it. If there is no enough space, you must resize your array. The execution results are as follows.

```
>./ex12-1 ↵
Enter a size first: 5 ↵
Enter a sequence (stop by -1):
6 3 5 2 1 -1
The medium is 3
>./ex12-1 ↵
Enter a size first: 5 ↵
Enter a sequence (stop by -1):
12 6 9 10 11 15 8 1 -1
The medium are 9 and 10.
>
```

- Please complete the following C/C++ program.

```
// file: ex12-1.cpp
#include <iostream>
```

```
using namespace std;

// function declaration
// InsertionSort() resize() and medium() are required

int main()
{
   int *array;
   int n;

   cout << "Enter a size first: ";
   cin >> n;

   array = new int [n];
   int x;

   while (cin >> x)
   {
      if (x != -1) break;

      // store x to array;
      // resize array if number of input is larger than n
   }
   // calculate the medium
  return 0;
}
void InsertionSort(int *vec,int size)
{
    for (int jdx = 1; jdx < size; jdx++)
    {
       int key = vec[jdx];
       int idx = jdx - 1;
       while ( idx >= 0 && vec[idx] > key)
       {
          vec[idx+1] = vec[idx];
          --idx;
       }
       vec[idx+1] = key;
    }
}
```

✓ Please write a program to read the information in file `ex12-2-in.txt` and output the statistical result to file `ex12-2.out`.

The format of ex12-2-in.txt is as follows. The first row indicates the number of students and the other rows list the detail information of each student. For each student, there are name, id, age and gender. You program have to read these information from file and store them to a dynamic structure array.

```
5
Ben 9613829 23 M
John 9813629 22 M
Marry 9713210 21 F
Jean 9811028 19 F
Tom 9823012 24 M
```

- Your program has to read the above information and store them in a dynamic structure array. You have to design a structure called student and use student *uee1302 = new student[num] to store each item, where num means the number of students. Moreover, delete [] uee1302 is required before the program finished.

- The output file of ex12-2-in.txt is as follows. Notice that, your program allows user to specific the output filename. If the user does not assign the filename, the default is test.txt.

```
There are: Ben, John, Marry, Jean and Tom
Their average age is 21.8 years old.
There are 3 boys and 2 girls.
```

- The execution results of program ex12-2 are as follows. User can use command line option –i or –o to specific the input or output filename, respectively.

```
> ./ex12-2
Usage: ./ex12-2 [-i] input_filename [-o] output_filename
> ./ex12-2 –i ex12-2-in.txt
Write output in test.txt file.
> ./ex12-2 –i ex12-2-a.in
File ex12-2-a.in does not exist!!
> ./ex12-2 –i ex12-2-in.txt –o ex12-2-out.txt
Write output in ex12-2-out.txt.
> ./ex12-2 –o ex12-2-out.txt –i ex12-2-in.txt
Write output in ex12-2-out.txt.
> ./ex12-2 –o ex12-2-out.txt
Input file is required.
```

## Exercise after LAB

✓ Please write a program to count the words in a sentence and search if the target word exists in the sentence. Note that each word can only be composed of `[0~9]` and `[a~z]`. Examples of the required format are given as follows.

```
>./ex12-3 ↵
Please enter a sentence:
http://www.google.com is an important web site in our life and the word
“google” even become a verb in used.
Please enter a target:
google
There are 22 words.
There are 2 “google” in this sentence.
>./ex12-3 ↵
Please enter a sentence:
<a href="./files/UEE1302_F09_LAB_10.pdf">LAB_10</a>
Please enter a target:
target
There are 11 words.
There is no “target” in this sentence.
>
```

✓ Please write a C/C++ program to display the sequence from 1 to ***n***. The user types ***n*** from the command line. There are also four options: ***-s, -d, -t, -q***. They denote *single*, *double*, *triple* and *quadruple* times of the sequence, respectively. Notice that `–s` is the default option. If the user types the undefined option or only type the program name, your program only shows the usage of the program. The execution results are as follows:

```
>./ex12-4
./ex12-4 [-d/-s/-t/-q] number
>./ex12-4 5 ↵
1 2 3 4 5
>./ex12-4 –s 5
1 2 3 4 5
>./ex12-4 –d 5
2 4 6 8 10
>./ex12-4 –t 5
3 6 9 12 15
>./ex12-4 –q 5
4 8 12 16 20
>./ex12-4 –a 5
./ex12-4 [-d/-s/-t/-q] number
```

➢ Hint: You can use `switch-case` statement to decide the option