UEE1302(1066) F12: Introduction to Computers and Programming

Lab 11: File I/O & Pointer

What you will learn from Lab 11

In this laboratory, you will understand how to use C++ ***pointers***.

TASK 11-1: FILE INPUT AND OUTPUT STREAM: FSTREAM

✓ Program `lab11-1` gives an example to using `fstream`, which can use the read and write mode at the same time.

```
// file: lab11-1.cpp
#include <fstream>
#include <iostream>
#include <cstdlib>
using namespace std;
int main(int argc, char *argv[])
{
    if (argc!=4)
    {
        cout << "Usage: ./lab10-6 <filename> <target> <replace>" << endl;
        return 1;
    }
    fstream out(argv[1], ios::in | ios::out);
    if (out.fail())
    {
        cout << "Cannot open the file " << argv[1] << endl;
        return 1;
    }

    out.seekp(atoi(argv[2]),ios::beg);
    out.put(*argv[3]);

    return 0;
}
```

- `ios::in | ios::`app means that the file can be operated in two modes.
- `return 1` indicates that the program terminates in a non-normal way. `exit(1)` is the same as `return 1`, but `exit(1)` can be used in the any return type of main function. For example, `exit(1)` can be used in `void main()`, but `return 1` cannot.

✓ Here below shows the sample result of executing program `lab10-6`

```
> ./lab11-1 lab10-4.txt 5 i
   Whin your program takes input from a file,
```

```
it is said to be reading from the file; when
your program sends output to a file, it is said
to be writing to the file.
new words
```

- First, the `seekp()` points to the character at position `5` in file `lab10-4.txt`. Later, the `put()` write the character `i` to the current position.
- In `seekp()`, there are three modes: `ios::beg`, `ios::cur` and `ios::end`. `ios::beg` indicates the stream points to the position at beginning of the file.

TASK 11-2 : POINTER

✓ A pointer is the memory address of a variable. For a variable `n`, there are two parts in this variable: the *value* and the *address*. Program `lab12-1` explains the concept of pointers.

```
//File: lab11-2.cpp
#include <iostream>
using namespace std;

int main()
{
    int n = 10;
    cout << "The value of n is " << n << endl;
    cout << "The address of n is " << &n << endl;
    return 0;
}
```

- `&n` takes the ***address*** of variable `n`.

✓ The pointer/address can be stored in a variable, called *pointer variable*. Program `lab11-3` introduces the usage of the pointer variables.

```
//File: lab11-3.cpp
#include <iostream>
using namespace std;

int main()
{
    int n = 10;
    int *p = &n;  // p is a pointer variable

    cout << "The address of n is " << &n << endl;
    cout << "The value of p is " << p << endl;
    cout << "The address of p is " << &p << endl;
    cout << "The variable pointed by p is " << *p << endl;
    return 0;
}
```

- `int *p = &n` means that the point variable `p` stores the address of variable `n`. Then, you can

observe that "`the address of n = the value of p`."

- `&p` means that the ***address*** of pointer variable `p`.
- `*p` means the variable ***pointed*** by p.

✓ Program `lab11-4` shows the example of pointer manipulations.

```
//File: lab11-4.cpp
#include <iostream>
using namespace std;

int main()
{
    int n = 40;
    int *p1 = 0, *p2 = 0;

    cout << "n = " << n << endl;

    p1 = &n;
    *p1 = 60;
    cout << "n = " << n << endl;

    p2 = p1;
    *p2 = 50;
    cout << "n = " << n << endl;

    return 0;
}
```

- Please modify the above program to show the correct result as follows.

```
> ./lab11-4
n = 40
n = 50
n = 60
```

✓ Program `lab11-5` below shows some examples of using for pointer manipulation including pointer declarations and assignments.

```
//File: lab11-5.cpp
#include <iostream>
using namespace std;

int main()
{
    int i = 10;
    int *pi  = 0;
    int *pi2 = &i;
```

```
    pi = pi2;
    pi2 = 0;
    pi = i;

    double d = 10.5;
    double *pd = &d;

    pi = pd;
    pi = &d;

    return 0;
}
```

- Please explain the reason of compile-time errors and fix them.
- &p denotes the address of pointer variable p.
- *p denotes the variable (data) pointed by p.

TASK 11-3 : NEW AND DELETE OPERATOR

- ***new*** operator (`new T ()`) is used to allocate a block of raw storage of *sizeof(T)* and ***delete*** operator is used to release the memory space allocated by ***new*** operator.
- Program `lab11-6` shows an example of using the ***new*** operator and ***delete*** operator

```
//File: lab11-6.cpp
#include <iostream>
using namespace std;

int main()
{
    int n = 10;
    int *p1 = new int;
    p1 = &n;

    int *p2 = new int(1024);

    cout << "*p1 = " << *p1 << endl;
    cout << "*p2 = " << *p2 << endl;

    delete p1;
    delete p2;

    return 0;
}
```

- `int *p = new int;` is used to allocate a memory space with size *sizeof(int)*
- `int *p = new int(1024);` is used to allocate a memory space with size *sizeof(int)* and assign the initial value to the memory space

TASK 11-4: DYNAMIC ARRAY

- ✓ Dynamic arrays allow various lengths of arrays.
- ✓ ***new*** operator (`new T ()`) is used to allocate a block of raw storage of *sizeof(T)*; `new T [n]` is used to allocate a block of raw storage of *n*sizeof (T)*
- ✓ ***delete*** operator is used to release the memory space allocated by ***new*** operator; `delete []` is used to release memory allocated by `new T[n]`
- ✓ Program `lab11-7` shows an example of using dynamic arrays.

```
//File: lab11-7.cpp
#include <iostream>
using namespace std;

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

    int *vec = new int[n];
    for (int idx = 0; idx < n; idx++)
    {
        vec[idx] = idx;
    }
    for (int idx = 0; idx < n; idx++)
    {
        cout << vec[idx] << " ";
    }
    cout << endl;

    delete []vec;
    return 0;
}
```

- ➢ It is good to release the memory space by ***delete*** operator.
- ➢ Again, `new T ()` is used to allocate a signal object; delete is used to release the memory allocated by `new T()`. `new T[n]` is used to allocate an array object; delete is used to release the space allocated by `new T[n]`.

TASK 11-5 : EXERCISES

✓ Please write a program to sum up two polynomials. Note that the maximum degree is 10. The required format is shown as follows.

```
>./ex11-1 ↵
Please enter the degree of polynomial 1: 4 ↵
Please enter the coefficient in order: 5 1 0 -2 6 ↵
Polynomial 1 is 5X^4 + X^3 - 2X + 6
Please enter the degree of polynomial 2: 4 ↵
Please enter the coefficient in order: 0 0 1 0 0 ↵
Polynomial 2 is X^2
Sum = 5X^4 + X^3 + X^2 - 2X + 6
>
```

```
>./ex11-1 ↵
Please enter the degree of polynomial 1: 4 ↵
Please enter the coefficient in order: 5 1 0 -2 6 ↵
Polynomial 1 is 5X^4 + X^3 - 2X + 6
Please enter the degree of polynomial 2: 3 ↵
Please enter the coefficient in order: 0 1 1 1 ↵
Polynomial 2 is X^2 + X + 1
Sum = 5X^4 + X^3 + X^2 - 1X + 7
>
```

Hint: the degree of polynomial 1 is larger than polynomial 2

✓ Please write a program to generate a sequence from 1 to n and arrange the values randomly. Notice that the elements in the sequence cannot repeat. An example of the required format is shown as follows.

```
>./ex11-2 ↵
Please enter a number: 10 ↵
6 9 8 3 1 4 5 10 2 7
Please enter a number: 20 ↵
9 10 11 19 3 8 1 16 5 4 7 12 14 13 17 20 15 2 6 18
Please enter a number: -1
>
```

Exercise after LAB

1. Test Score

Write a program that dynamically allocates an array large enough to hold a user-defined number of test scores. Once all the scores are entered, the array should be passed to a function that sorts them in ascending order. Another function should be called that calculates the average score. The program should display the sorted list of scores and averages with appropriate headings. Use pointer notation rather than array notation whenever possible.
Input validation: Do not accept negative numbers for test scores.

2. Median Function

In statistics the median of a set of values is the value that lies in the middle when the values are arranged in sorted order. If the set has an even number of values, then the median is taken to be the average of the two middle values. Write a function that determines the median of a sorted array. The function should take and array of numbers and an integer indication the size of the array and return the median of the values in the array. You may assume the array is already sorted. Use pointer notation whenever possible.

3. Movie statistics

Write a program that can be used to gather statistical data about the number of movies college students see in a month. The program should ask the user how many students were surveyed and dynamically allocate an array of that size. The program should then allow the user to enter the number of movies each student has seen. The program should than calculate the average, median, and mode of the values entered.