Week 10

Arrays
C++ provides a data structure, the array, which stores a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

Array Declaration:
In C++, arrays are declared and accessed by the use of square brackets.

E.g.:
(a) One-dimension array
int num[10];

	0	1	2	3	4	5	6	7	8	9
num										
				num[3]					num[8]	

(b) Two-dimension array
float matrix[5][4];

matrix	0	1	2	3
0				
1		matrix[1][1]		
2				matrix[2][3]
3				
4				

C++ has no bounds checking, so the compiler will let you refer to array elements even if they are outside the declared range.
For example, we could refer to num[20].
The moral is, be very careful never to use an array element which has not been explicitly declared as part of the allowed range.

Initializing Array:
We may put values into any individual element of an array using the assignment operator.
E.g.: int num[10]; num[4]=99;
int num[10]={3,99,2,-4,6,29,84,4,-7,25};
int num[]={3,99,2,-4,6,29,84,4,-7,25}; //The compiler can automatically size the array
If we try to display a numeric array using cout (i.e. cout<<num;) then we will see that what is displayed is the address of the first element of the array.

Example:
```
#include <iostream>

int foo [] = {16, 2, 77, 40, 12071};
int n, result=0;

int main ()
{
        for ( n=0 ; n<5 ; ++n )
        {
                result += foo[n];
        }
        cout << result;
        return 0;
}
```

Arrays and Functions:

Arrays are always implicitly passed by reference to functions, never by value.
They cannot be used as the return type of a function.

Example:

```
#include <iostream.h>

void setArraySize(int size, int array_in[])
{
        array_in[0]=size;
}

void main()
{
        const int ARRAY_SIZE=15;
        int an_array[ARRAY_SIZE];
        setArraySize(ARRAY_SIZE,an_array);
        cout<<"The first element is "<<an_array[0]<<endl;
}
```

Output:

```
The first element is 15
```

Example:

```
#include <iostream>

void printarray (int arg[], int length)
{
        for (int n=0; n<length; ++n)
        cout << arg[n] << ' ';
        cout << '\n';
}

int main ()
{
        int firstarray[] = {5, 10, 15};
        int secondarray[] = {2, 4, 6, 8, 10};
        printarray (firstarray,3);
        printarray (secondarray,5);
}
```

Structures (struct)
A Structure is a collection of related data items, possibly of different types. A structure type in C++ is called struct. A struct is heterogeneous in that it can be composed of data of different types. In contrast, array is homogeneous since it can contain only data of the same type. Structures hold data that belong together.
Examples:
Student record: student id, name, major, gender, start year, …
Bank account: account number, name, currency, balance, …
Address book: name, address, telephone number, …

A struct is named as a whole while individual members are named using field identifiers. Complex data structures can be formed by defining arrays of structs.

Structure Declaration:

```
struct <struct-type>
{
        <type> <identifier_list>;
        <type> <identifier_list>;
        ...
} ;
```

Example:

```
struct date
{
        int day;
        int month;
        int year;
} ;
```

Example:

```
struct studentInfo
{
        int id;
        int age;
        char gender;
        double cga;
};
```

Example Programs:

Example:

```
#include <iostream.h>

struct person
{
        int id;
        char gender;
        float salary;
};

void main()
{
        person p1;
        cout << "Enter id: ";
        cin>>p1.id;
        cout << "Enter gender: ";
        cin >> p1.gender;
        cout << "Enter salary: ";
        cin >> p1.salary;
        cout << "\nDisplaying Information." << endl;
        cout << "Id: " << p1.id << endl;
        cout <<"Gender: " << p1.gender << endl;
        cout << "Salary: " << p1.salary;
}
```

Example:

```
#include<iostream.h>

struct employee
{
        int id;
        int age;
        long salary;
};

void main()
{
        int i;
        employee emp[ 3 ];

        for(i=0;i<3;i++)
        {
                cout << "\nEnter details of the Employee"<< i+1;
                cout << "\n\tEnter Employee Id : "; cin >> emp[i].id;
                cout << "\n\tEnter Employee Age : "; cin >> emp[i].age;
                cout << "\n\tEnter Employee Salary : "; cin >> emp[i].salary;
        }
        cout << "\nDetails of Employees";
        for(i=0;i<3;i++)
                cout << "\n"<< emp[i].id <<"\t" << emp[i].age <<"\t"<< emp[i].salary;
    }
```

Output:

```
Enter details of Employee1
        Enter Employee Id : 101
        Enter Employee Age : 29
```

```
          Enter Employee Salary : 45000

Enter details of Employee2
          Enter Employee Id : 102
          Enter Employee Age : 31
          Enter Employee Salary : 51000

Enter details of Employee3
          Enter Employee Id : 103
          Enter Employee Age : 28
          Enter Employee Salary : 47000

Details of Employees
101      29     45000
102      31     51000
103      28     47000
```