C/C++ Program Design

LAB 3

CONTENTS

- Learn how to create and use arrays(Declare, Initialize and Access)
- Master character arrays and strings
- Learn how to create and use structures(Declare, Initialize and Access)

2 Knowledge Points

2.1 Array

2.2 Character arrays and strings

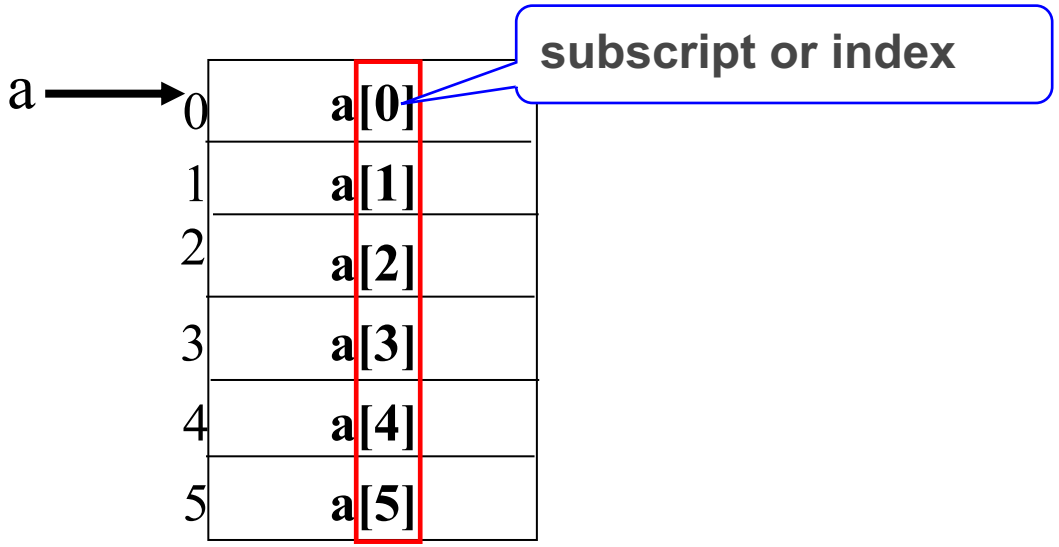
2.3 Structure

2.1 Array

- Arrays are fixed-size collections consisting of data items of the same type.
- The index of an array is from 0.
- C++ compiler does not report whether the array is out-off-bounds.

One-dimension array

int a[6];



lab03_examples > onedarray.cpp > ...

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

int main()
{
    int foo[] = {16, 2, 77, 40, 12071};
    int a = 1;

    foo[0] = a;
    foo[1] = -34;
    a = foo[2];

    cout << "foo[0] = " << foo[0] << endl;
    cout << "foo[1] = " << foo[1] << endl;
    cout << "foo[2] = " << foo[2] << endl;
    cout << "a = " << a << endl;

    cout << "The size of foo is:" << sizeof(foo) << " bytes" << endl;
    cout << "The size of an element of foo is:" << sizeof(foo[0]) << " bytes" << endl;
    cout << "There are " << sizeof(foo)/sizeof(foo[0]) << " elements in foo." << endl;

    return 0;
}
```

Define and initialize a one-dimension array

Use [ ] operator to access the elements of the array

The array index starts from 0

Use the **sizeof** operator with an **array name**, you'll get the number of bytes in the whole array.

Use **sizeof** with an **array element**, you'll get the size, in bytes, of the element.

Count the number of elements in foo

```
foo[0] = 1
foo[1] = -34
foo[2] = 77
a = 77
The size of foo is:20 bytes
The size of an element of foo is:4 bytes
There are 5 elements in foo.
```

lab03_examples > arrayini.cpp > ...

```cpp
#include <iostream>
#include<iomanip>
#define SIZE 2
const int arrsize = 3;

int main()
{
    using std::cout;
    using std::endl;
    int a[SIZE] = {0};
    double b[arrsize] = {1};

//    cout << std::fixed;
    cout << "The elements in a are:" << a[0] << "," << a[1] << endl;
    cout << "The elements in b are:" << b[0] << "," << b[1] << "," << b[2] << endl;

    return 0;

}
```

Use preprocessor **#define** to define a symbolic constant

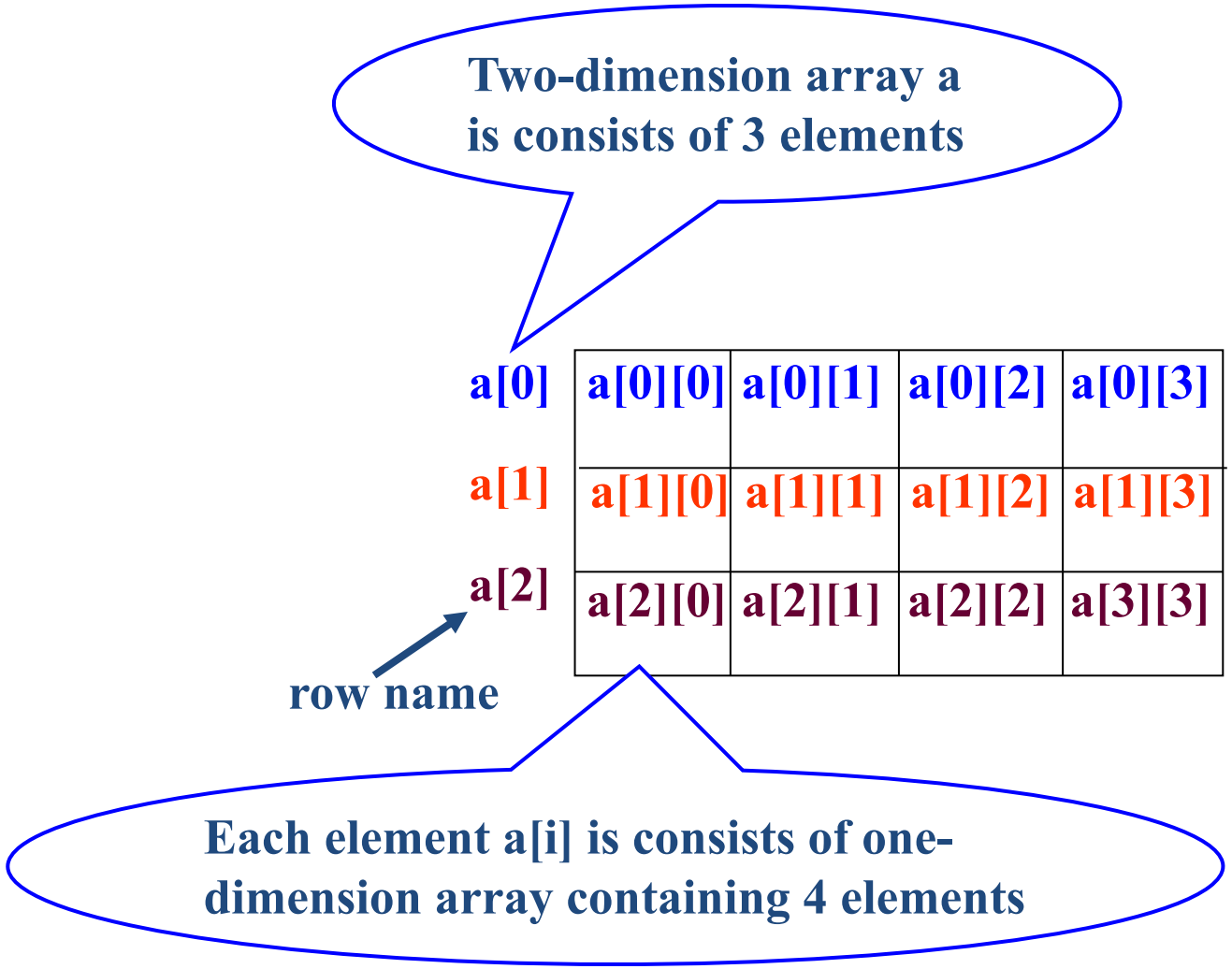
Use C++ style to define a constant

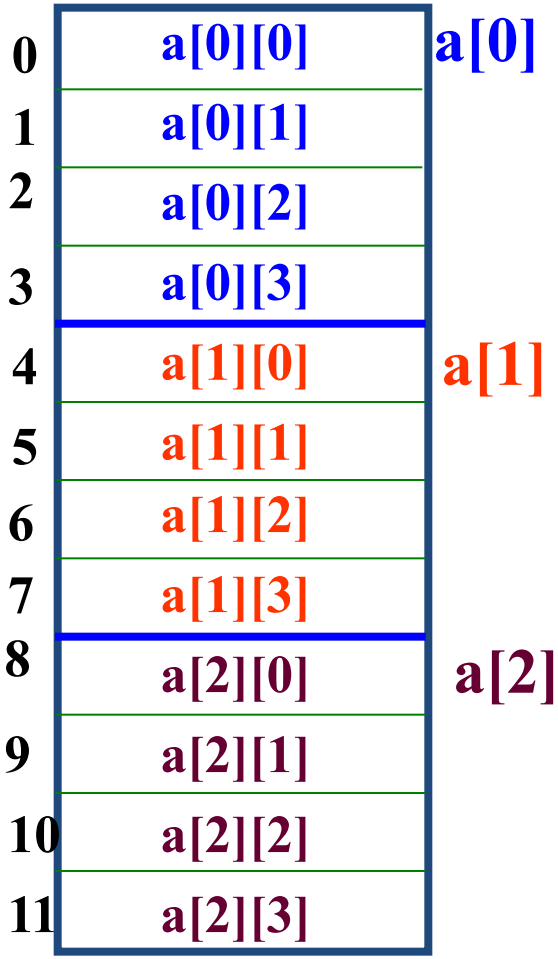
Initialize an array partially, the remaining elements are 0. The number of elements must be specified.

```
The elements in a are:0,0
The elements in b are:1,0,0
```

Two-dimension array

int a[3][4];





lab03_examples > twodarray.cpp > main()

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

int main()
{
    int test[3][2] =
    {
        {2, -5},
        {4, 0},
        {9, 1}
    };
    //Accessing two dimensional array
    cout << "test[0][1] = " << test[0][1] << endl;
    cout << "test[2][0] = " << test[2][0] << endl;

    cout << "The size of test is:" << sizeof(test) << endl;
    cout << "The size of the first row of test is:" << sizeof(test[0]) << endl;
    cout << "The size of an element in test is:" << sizeof(test[0][0]) << endl;

    return 0;

}
```

Define and initialize a two-dimension array

Use [ ] [ ]operator to access the elements of the array

```
test[0][1] = -5
test[2][0] = 9
The size of test is:24
The size of the first row of test is:8
The size of an element in test is:4
```

2.2 Character array and strings

2.2.1 Define a C-string

You can use one of the four ways below to define a character array:

```
char str[ ] = “C++” ;
char str[4] = “C++” ;
char str[ ] = {‘C’, ‘+’, ‘+’, ‘\0’};
char str[4] = {‘C’, ‘+’, ‘+’, ‘\0’}
```

Strings end with \0

2.2.2 string class

string is a class of C++, it can be used as a type.

```
string str1 = “C and C++”;   //initialize str1 with a string

string str2 = str1 + “programming”;  //use + to concatenate two or more strings
```

2.2.3 Keyboard input and terminal output of character array

1. C: scanf & printf
   - %d ----int
   - %f ----float
   - %c -----char
   - %s -----string

lab03_examples > C scanf_printf.c > ...

```c
#include <stdio.h>

int main()
{
    char str[20];
    printf("Enter a string:\n");
    scanf("%s", str);
    printf("You entered: %s\n",str);

    return 0;

}
```

Why only Computer?

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ gcc scanf_printf.c
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ ./a.out
Enter a string:
Computer
You entered: Computer
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ ./a.out
Enter a string:
Computer Science
You entered: Computer
```

scanf uses whitespace—spaces, tabs, and newlines to delineate a string.

2. C: gets & puts

```
fgets(str, 20, stdin);
```

gets_puts.c > ...

```c
#include <stdio.h>

int main()
{
    char str[20];
    printf("Enter a string:\n");
    gets(str);
    printf("You entered: ");
    puts(str);

    return 0;
}
```

Use gets function to gain the whole line

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ gcc gets_puts.c
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ ./a.out
Enter a string:
Computer Science
You entered: Computer Science
```

scanf()
when *scanf()* is used to read string input it stops reading when it encounters **whitespace, newline** or **End Of File**
It is used to read input of **any datatype**

gets()
when *gets()* is used to read input it stops reading input when it encounters **newline** or **End Of File.** It does not stop reading the input on encountering whitespace as it considers whitespace as a string.
It is used **only for string** input.

3. C++: cin & cout

lab03_examples > cin_cout.cpp > ...

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

int main()
{
    char str[100];

    cout << "Enter a string:";
    cin >> str;
    cout << "You entered: " << str << endl;

    cout << "Enter an other string:";
    cin >> str;
    cout << "You entered: " << str << endl;

    return 0;
}
```

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ g++ cin_cout.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ ./a.out
Enter a string:C++
You entered: C++
Enter an other string:programming is fun.
You entered: programming
```

The cin is to use whitespace-- spaces, tabs, and newlines to delineate a string.

4. C++: cin.get( )

Input a single character:
istream& get(char&);
int get(void);

Input a string:
istream& get(char*,int);

lab03_examples > cin_get.cpp > ...

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

int main()
{
    char str[20];

    cout << "Enter a string:";
    cin.get(str, 20);
    cout << "You entered: " << str << endl;

    cin.get();
    cout << "Enter an other string:";
    cin.get(str, 20);
    cout << "You entered: " << str << endl;

    return 0;
}
```

If the statement is omitted, what will be the output?

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ g++ cin_get.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ ./a.out
Enter a string:Programming is fun.
You entered: Programming is fun.
Enter an other string:C/C++ programming is fun.
You entered: C/C++ programming i
```

If the length of input string is greater than 20, it can only store first 19 characters in str.

5. C++: cin.getline( )

Input a string:

istream& getline(char*,int);

lab03_examples > cin_getline.cpp > ...

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

int main()
{
    char str[20];

    cout << "Enter a string:";
    cin.getline(str, 20);
    cout << "You entered: " << str << endl;

    cout << "Enter an other string:";
    cin.getline(str, 20);
    cout << "You entered: " << str << endl;

    return 0;
}
```

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ g++ cin_getline.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ ./a.out
Enter a string:Programming is fun.
You entered: Programming is fun.
Enter an other string:C/C++ programming is fun.
You entered: C/C++ programming i
```

If the length of input string is greater than 20, it can only store first 19 characters in str.

cin.get( ) vs cin.getline( )

getline() and get() both read an entire input line—that is, up until a newline character. However, getline() discard the newline character, whereas get() leave it in the input queue.

lab03_examples > get_getline.cpp > ...

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

int main()
{
    char str[20];

    cout << "Enter a string:";
    cin.get(str, 20);
    cout << "You entered: " << str << endl;

    cout << "Enter an other string:";
    cin.getline(str, 20);
    cout << "You entered: " << str << endl;

    return 0;
}
```

Program runs without entering another string

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ g++ get_getline.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ ./a.out
Enter a string:C and C++
You entered: C and C++
Enter an other string:You entered:
```

lab03_examples > get_getline.cpp > main()

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

int main()
{
    char str[20];

    cout << "Enter a string:";
    cin.get(str, 20);
    cout << "You entered: " << str << endl;

    cin.get();
    cout << "Enter an other string:";
    cin.getline(str, 20);
    cout << "You entered: " << str << endl;

    return 0;
}
```

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ g++ get_getline.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ ./a.out
Enter a string:C and C++
You entered: C and C++
Enter an other string:Programming is fun.
You entered: Programming is fun.
```

6. string class I/O

getline() function takes the input stream as the first parameter which is cin and str as the location of the line to be stored.

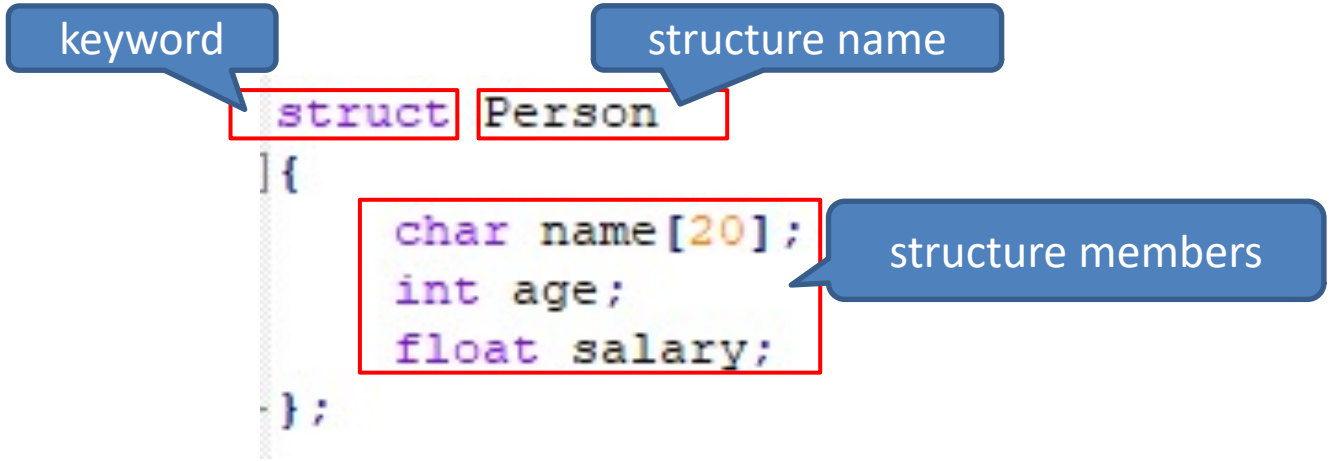
lab03_examples > string.cpp > ...

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

int main()
{
    string str;
    cout << "Enter a string:";
    getline(cin, str);
    cout << "You entered: " << str << endl;

    cout << "Enter another string:";
    getline(cin,str);
    cout << "You entered: " << str << endl;

    return 0;
}
```

```
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ g++ string.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ ./a.out
Enter a string:Programming is fun.
You entered: Programming is fun.
Enter another string:C/C++ programming is fun.
You entered: C/C++ programming is fun.
```

2.3 Structure

2.3.1 Declare a structure



When a structure is declared, no memory is allocated.

2.3.2 Define, initialize and access a structure variable

lab03_examples > structureini.cpp > main()

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

struct Person  //structure declaration
{
    char name[20];
    int age;
    float salary;
};

int main()
{
    Person p1 = {
        "Glorious Gloria",     //name value
        23,      //age value
        1034.9    //salary value
    };

    cout << "\nDisplaying Information of p1:" << endl;
    cout << "Name: " << p1.name << endl;
    cout << "Age: " << p1.age << endl;
    cout << "Salary: " << p1.salary << endl;

    return 0;
}
```

Declare a structure

Define and initialize a structure variable

Access a structure members use . operator

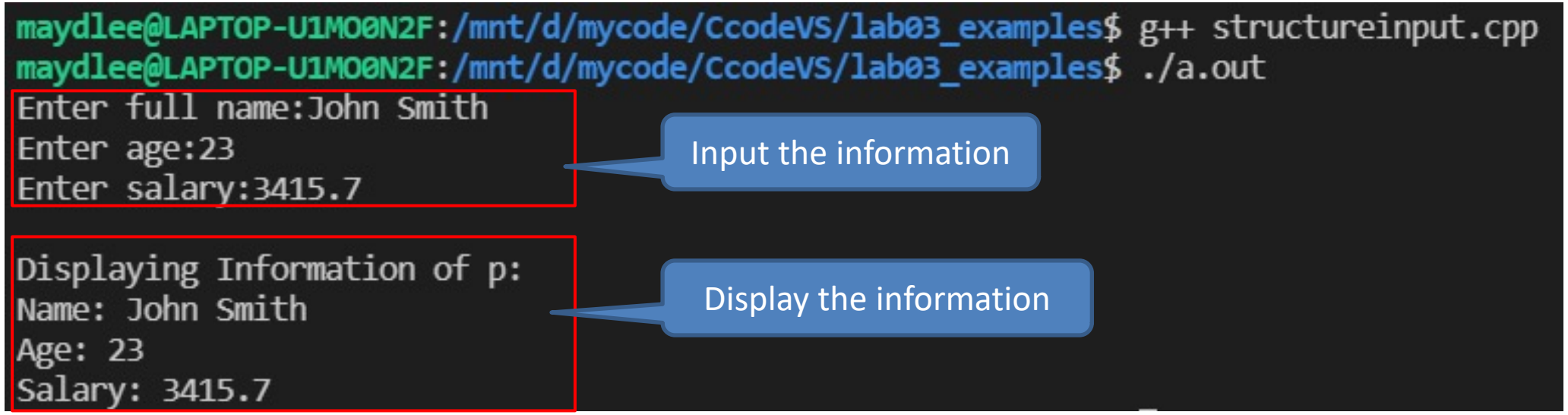
```
Displaying Information of p1:
Name: Glorious Gloria
Age: 23
Salary: 1034.9
```

Input data from keyboard

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

struct Person  //structure declaration
{
    char name[20];
    int age;
    float salary;
};

int main()
{
    Person p;

    cout << "Enter full name:";
    cin.get(p.name, 20);
    cout << "Enter age:";
    cin >> p.age;
    cout << "Enter salary:";
    cin >> p.salary;

    cout << "\nDisplaying Information of p:" << endl;
    cout << "Name: " << p.name << endl;
    cout << "Age: " << p.age << endl;
    cout << "Salary: " << p.salary << endl;

    return 0;
}
```

Define a structure variable

Input data from keyboard to the structure members


maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ g++ structureinput.cpp
maydlee@LAPTOP-U1MO0N2F:/mnt/d/mycode/CcodeVS/lab03_examples$ ./a.out
Enter full name:John Smith
Enter age:23
Enter salary:3415.7
Input the information
Displaying Information of p:
Name: John Smith
Age: 23
Salary: 3415.7
Display the information

2.3.2 Other operations of structure

lab03_examples > person.h > ...

```
1  typedef struct Person  //str
2  {
3      char name[25];
4      int age;
5      float salary;
6  }PERSON;
```

Use preprocessor **#typedef** to declare an alias of structure type

lab03_examples > structureop.cpp > ...

```
1  #include <iostream>
2  #include "person.h"
3  using namespace std;
4
5  int main()
6  {
7      Person p1 = {"Alice Green", 34,2902.5};
8      PERSON p2;
9
10     cout << "The size of Person is:" << sizeof(Person) << endl;
11     cout << "The size of p1 is:" << sizeof p1 << endl;
12
13     cout << "Structure assignment:" << endl;
14     p2 = p1;
15     cout << "Information of P2:" << endl;
16     cout << "name is:" << p2.name << endl;
17     cout << "age is:" << p2.age << endl;
18     cout << "salary is:" << p2.salary << endl;
19
20     return 0;
21 }
```

You can use structure name or alias to define structure variables

You can assign a structure variable to another one with the same structure type

```
The size of Person is:36
The size of p1 is:36
Structure assignment:
Information of P2:
name is:Alice Green
age is:34
salary is:2902.5
```

Normally, the size of a structure is not the sum of its member lengths

2.3.3 Array of Structure

lab03_examples > structurearray.cpp > ...

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

struct Employee
{
    string name;
    int age;
};

int main()
{
    Employee struArray[3];

    struArray[0].name = "Harvey";
    struArray[0].age = 33;
    struArray[1].name = "Sally";
    struArray[1].age = 26;
    struArray[2].name = "Jeff";
    struArray[2].age = 52;

    cout << "Displaying the Array Contents" << endl;
    for(int i = 0; i < 3; i++)
        cout << "Name: " << struArray[i].name << "\tAge: " << struArray[i].age << endl;

    cout << "The size of strucutre is:" << sizeof(Employee) << endl;
    cout << "The size of structure array is:"  << sizeof(struArray)  << endl;
    cout << "The size of name is:" << sizeof(struArray[0].name) << endl;

    return 0;
}
```

Declare a structure

Define a structure array

Access the elements of structure array

```
Displaying the Array Contents
Name: Harvey    Age: 33
Name: Sally     Age: 26
Name: Jeff      Age: 52
The size of strucutre is:40
The size of structure array is:120
The size of name is:32
```

3 Exercises

1. Write a C++ program that asks the user to input the information, as shown in the following example of output:

```
What is your first name? Betty Sue
What is your last name? Yewe
What letter grade do you deserve? B
What is your age? 20
The information you entered is:
Name: Betty Sue, Yewe
Grade: B
Age: 20
```

Note that the program should be able to accept first name that comprise more than one word.

2. The **CandyBar** structure contains **three** members. The first member holds the brand **name** of candy bar. The second member holds the **weight**(which may have a fractional part) of the candy bar, and the third member holds **the number of calories**(an integer value) in the candy bar.

Write a program that uses a structure variable of that type. The program should ask the user to enter each of the preceding items of information, and then the program should display that information. Use **cin**(or its methods) and **cout**.

Sample output:

```
Enter brand name of a Candy bar: Mocha Munch
Enter weight of he Candy bar: 2.3
Enter calories (an integer value) in the Candy bar: 350
Brand: Mocha Munch
Weight: 2.3
Calories: 350
```

3. The **CandyBar** structure contains three members, as described in **Exercise 2**. Write a program that creates an array of **three CandyBar** structures, initializes them to value of your input, and then displays the contents of each structure. Find the greatest calories per weight, display the name and calories per weight of which satisfies the condition.

Sample output:

```
Please input three CandyBar's information:
Enter the brand name of candy bar: Ferro Rocher
Enter the weight: 23.6
Enter the calories: 893
Enter the brand name of candy bar: Hershey's
Enter the weight: 13.2
Enter the calories: 658
Enter the brand name of candy bar: Mars Wrigley
Enter the weight: 3.2
Enter the calories: 127

Displaying the CandyBar array contents
Brand name: Ferro Rocher
Weight: 23.6
Calories: 893
Brand name: Hershey's
Weight: 13.2
Calories: 658
Brand name: Mars Wrigley
Weight: 3.2
Calories: 127

The greatest calories per weight is:
Brand name: Hershey's
Calories per weight:49.8485
```