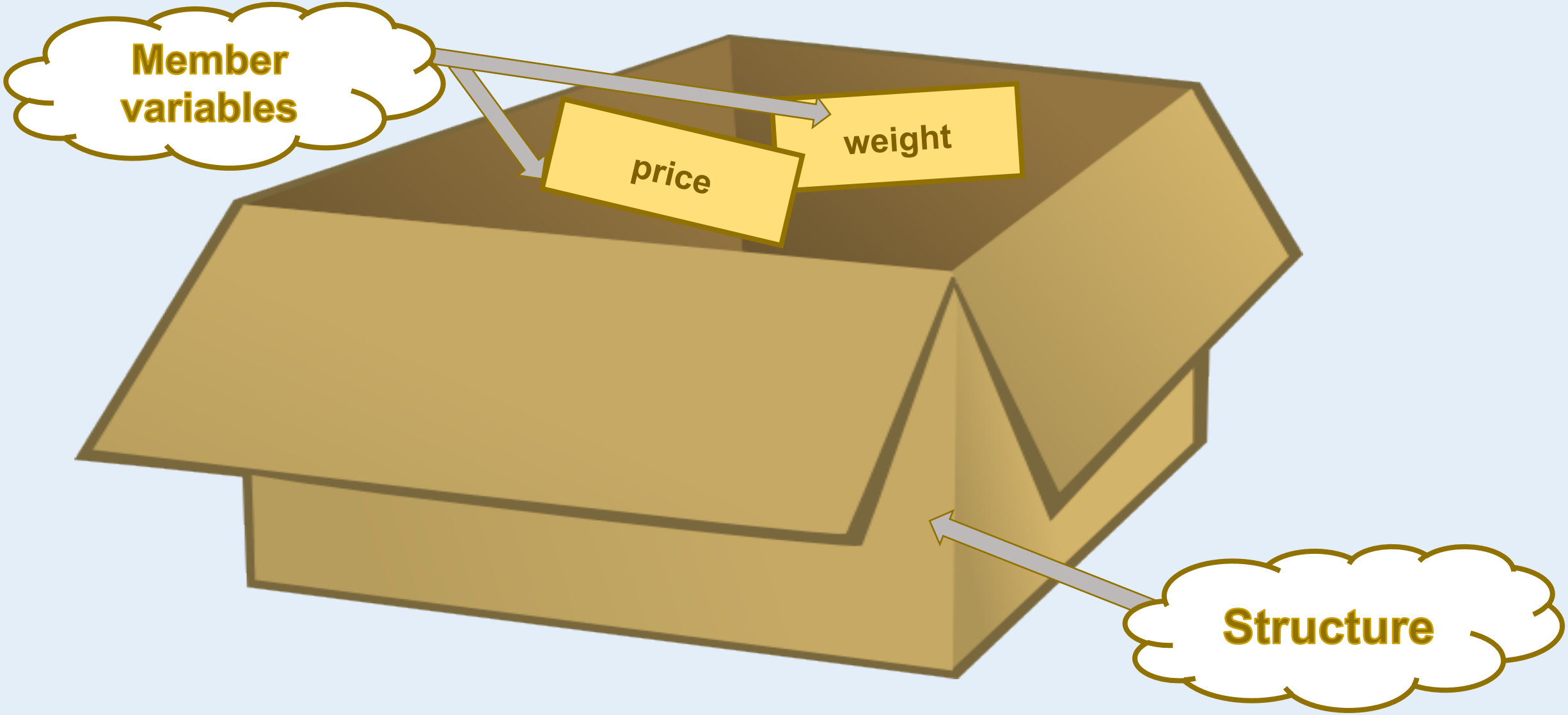
Lab #4

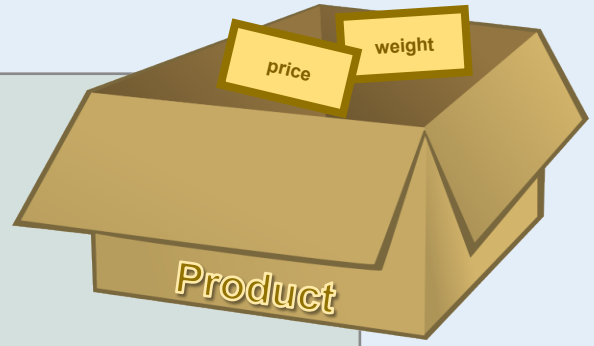
Structures

Structured Programming 2016/2017

What is a Structure? Why?



Declaration and Initialization




```
struct product
{
        int weight;
        float price;
}apples, bananas;

int main() {
product gold = {};

product bagOfFlour =  { 1 , 10 }, bagOfSugar;
…
}
```

Referencing Member Variables

Dot Operator

```
product bagOfFlour;

bagOfFlour.weight = 10;

cin>>bagOfFlour.price;

cout<<“Flour weight = ”<< bagOfFlour.weight<<
“, price = “<< bagOfFlour.price;
```

What is wrong with the following C declarations?

A. **struct point (double x, y)**

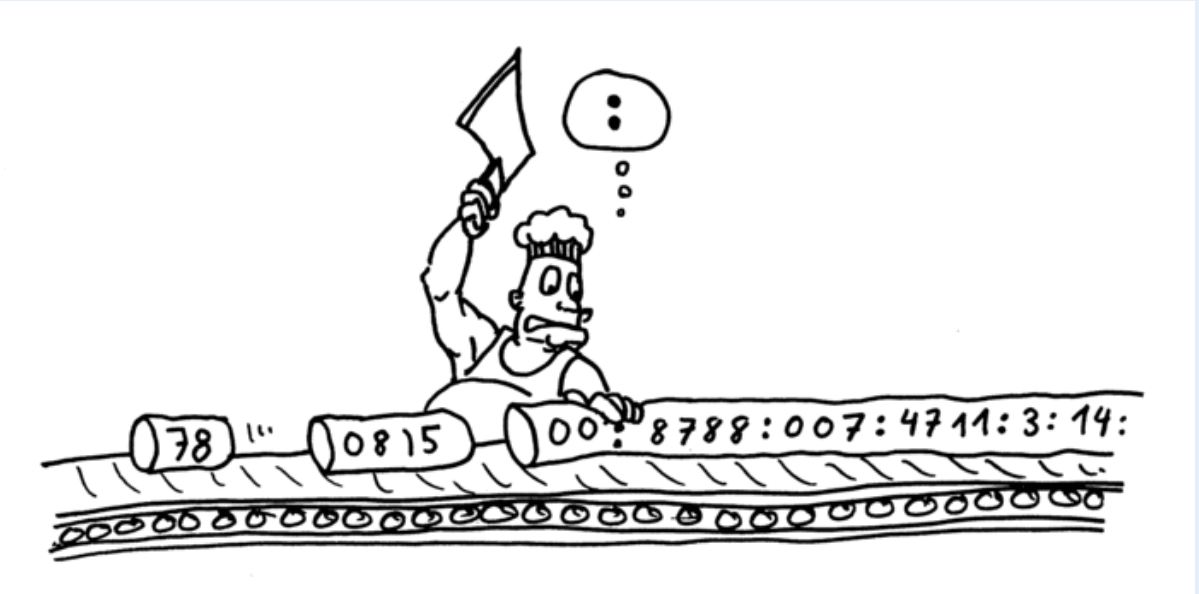
B. **struct point { double x, double y };**

C. **struct point { double x; double y }**

D. **struct point { double x; double y; };**

E. **struct point { double x; double y; }**

Exercises!



Representing Distances

Write a program that represents the **distance** in the form of a **structure** containing **meters**, and **centimeters**.

Define 3 variables and input their values, as follows:

- The first value from the user.
- The second value from Assigning Values.
- The third value is the result of addition of the other two values.

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

// Define distance structure
struct dist
{
    int meters;
    int centimeters;
};
```

```
int main()
{
    dist d1, d2, d3;

    // Enter the values of the first variable members
    cout << "Enter the values of the first variable: meters then
    centimeters" <<endl;
    cin >> d1.meters >> d1.centimeters;

    // Assign Values of the second variable members
    d2.meters = 15;
    d2.centimeters = 40;

    // Add the first and second variable members and put the result in the
    third variable
    d3.meters = d1.meters + d2.meters;
    d3.centimeters = d1.centimeters + d2.centimeters;
```

```
    // Put the third variable in the correct meters and centimeters format
    while (d3.centimeters >= 100)
    {
        d3.meters++;
        d3.centimeters -= 100;
    }

    //Display the third variable members
    cout<< d1.meters << "." << d1.centimeters << " + "
    << d2.meters << "." << d2.centimeters << " = "
    << d3.meters << "." << d3.centimeters << endl;

    return 0;
}
```

Ready… Steady… Code!

1. Date Struct

Write a program that represent Date as a structure. Each Day, Month and Year. Write a program that reads from the user the date and Display it in the format 21/11/1991

Sample Execution:

```
Enter Day : 21
Enter Month : 9
Enter Year : 1991

The Date you entered is: 21/11/1991
```

```
#include <iostream>
using namespace std;
struct Date
{
	int Day;
	int Month;
	int Year;
};
```

```
void main()
{

	Date MyDate;
	cout << "Enter Day : ";
	cin >> MyDate.Day;
	cout << "Enter Month : ";
	cin >> MyDate.Month;
	cout << "Enter Year : ";
	cin >> MyDate.Year;

	cout << "The Date you entered is: " << MyDate.Day <<
"/" << MyDate.Month << "/" << MyDate.Year<<endl;

}
```

2. Calculate Student Total Grade

Write a program that represent Student as a structure. Each student has ID, Name, Date of birth (the date from previous exercise), 6 Grades and Total Grade, Write a program that reads from the user the data of student and calculate the total grade

Sample Execution:

```
Enter Student ID : 1
Enter Student Name : Ahmed
Enter Date of birth: 21 11 1991
Enter 6 grades for the student : 20 10 60 30 50 40

Total Grade is : 210
```

```cpp
#include<iostream>
using namespace std;
struct Date
{
    int Day;
    int Month;
    int Year;
};

// Define distance structure
struct Student
{
    int ID;
    char Name[10];
    int Grades[6];
    int Total=0;
    Date DateOfBirth;
};
```

```
int main()
{
    Student S;

    cout << “Enter student Id : " <<endl;
    cin >> S.Id;

    cout << “Enter student Name : " <<endl;
    cin >> S.Name;

    cout << “Enter student Date of birth : " <<endl;
    cin >> S.DateOfBirth.Day>> S.DateOfBirth.Month>> S.DateOfBirth.Year;

    cout << “Enter 6 grades for the student :" <<endl;
    for (int i = 0; i < 6; i++)
    {
        cin >> S.Grades[i];
        S.Total += S.Grades[i];
    }
    cout << "Student Total Grade is : " << S.Total;
}
```

3. Calculate Students Total Grade

Modify the previous program to reads and stores the data of **Several Students** and calculates the total of each student.

3. Total Per students

Sample Execution:

```
Enter how many student you want to enter : 3

Enter ID for Student 1 : 100
Enter Name for Student 1 : Ahmed
Enter Date of birth for student 1 : 21 11 1991
Enter 6 grades for student 1 : 10 20 10 20 10 20

Enter ID for Student 2 : 101
Enter Name for Student 2 : Mohamed
Enter Date of birth for student 2 : 20 9 1991
Enter 6 grades for student 2 : 60 50 40 60 50 40
```

Total Per students

Sample Execution:

```
Enter ID for Student 3 : 102
Enter Name for Student 3 : Kamal
Enter Date of birth for student 3 : 15 11 1991
Enter 6 grades for student 3 : 60 30 50 20 40 10

The Total For Student 1 : 90
The Total For Student 2 : 300
The Total For Student 3 : 210
```

```cpp
#include<iostream>
using namespace std;
struct Date
{
    int Day;
    int Month;
    int Year;
};

// Define distance structure
struct Student
{
    int ID;
    char Name[10];
    int Grades[6];
    int Total=0;
    Date DateOfBirth;
};
```

```
int main()
{
    Student S[10];
    cout << “Enter how many students you will enter : " <<endl;
    int n;
    cin >> n;

    for( int j =0; j<n; j++)
    {
        cout << “Enter student Id of student # " << j+1 <<endl;
        cin >> S[j].Id;

        cout << “Enter student Name of student # " << j+1 <<endl;
        cin >> S[j].Name;
        cout << “Enter 6 grades for the student # " << j+1 <<endl;
        for (int i = 0; i < 6; i++)
        {
                cin >> S[j].Grades[i];
                S[j].Total += S[j].Grades[i];
        }
    }
```

```
    for (int i = 0; i < 6; i++)
    {
            cout << “The Total Grade for the student # " << i+1 <<
                 “ is " << S[i].Total;

    }
}
```

4. Student with max grade

Modify the previous program to get the student with the max total and display his data.

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

// Define distance structure
struct Student
{
    int ID;
    char Name[10];
    int Grades[6];
    int Total=0;
};
```

```
int main()
{
    Student S[10];
    cout << “Enter how many students you will enter : " <<endl;
    int n;
    cin >> n;

    for( int j =0; j<n; j++)
    {
        cout << “Enter student Id of student # " << j+1 <<endl;
        cin >> S[j].Id;

        cout << “Enter student Name of student # " << j+1 <<endl;
        cin >> S[j].Name;
        cout << “Enter 6 grades for the student # " << j+1 <<endl;
        for (int i = 0; i < 6; i++)
        {
                cin >> S[j].Grades[i];
                S[j].Total += S[j].Grades[i];
        }
    }
```

```
int max= S[0].Total, maxindex =0;
for (int i = 0; i < 6; i++)
{
        cout << “The Total Grade for the student # " << i+1 <<
             “ is " << S[i].Total;
      if(S[i].Total >max)
      {
            max=S[i].Total;
            maxindex=i;
      }
}
cout << “Student with the max total is :“ <<endl;
cout<< “ID : ” << S[maxindex].Id<<endl;
cout<< “Name : ” << S[maxindex].Name<<endl;
cout<< “Total : ” << S[maxindex].Total<<endl;

}
```

Part Cost

A vendor is buying from a **part**, which is represented by

- The Part number: PNum
- The Part name: PName
- The Quantity the vendor bought from this part: PQ
- The Price of this part: PP

Write a program that reads from the user and stores the data of this part bought and calculates how much the vendor should pay.

Part Cost

Sample Execution:

```
Enter Part Number:16
Part Name: ports
Part Quantity: 5
Part Price: 5

Total Pay = 25
```

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

// Define Part structure
struct Part
{
	int PNum;
	char PName[50];
	int PQ;
	float PP;
};
```

```
int main()
{
	float TotalPay=0;
	Part X;
	// Enter the Part members
	cout << "Enter Part Number" <<endl;
	cin >>X.PNum;
	cout << "Enter Part Name" <<endl;
	cin >>X.PName;
	cout << "Enter Part Quantity" <<endl;
	cin >>X.PQ;
	cout << "Enter Part Price" <<endl;
	cin >>X.PP;
	//Calculate and Display Total Pay
	TotalPay=X.PP * X.PQ;
	cout<< "Total Pay ="<<TotalPay<<endl;
	return 0;
}
```

Parts Cost

Modify the previous program to reads and stores the data of **Several parts** bought and calculates how much the vendor should pay for those bought parts. (Assume the vendor bought Five parts).

Parts Cost

Sample Execution:

```
Enter Part Number:12
Part Name: Screw
Part Quantity: 5
Part Price: 5
Enter Part Number:13
Part Name: hard
Part Quantity: 4
Part Price: 4
Enter Part Number:14
Part Name: IC
Part Quantity: 3
Part Price: 3
```

```
Enter Part Number:15
Part Name: wires
Part Quantity: 2
Part Price: 2
Enter Part Number:16
Part Name: ports
Part Quantity: 1
Part Price: 1

Total Pay = 55
```

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

// Define Part structure
struct Part
{
	int PNum;
	char PName[50];
	int PQ;
	float PP;
};
```

```
int main()
{       const int size=5;
        float TotalPay=0;
        Part arr[size];

        // Enter the List of Parts members
        for(int i=0;i<size;i++)
        {
                        cout << "Enter Part Number" <<endl;
                        cin >>arr[i].PNum;
                        cout << "Enter Part Name" <<endl;
                        cin >>arr[i].PName;
                        cout << "Enter Part Quantity" <<endl;
                        cin >>arr[i].PQ;
                        cout << "Enter Part Price" <<endl;
                        cin >>arr[i].PP;
        }
```

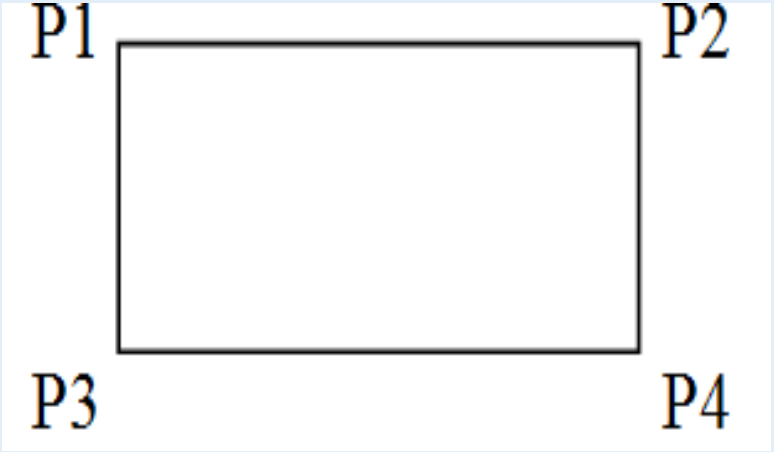
```
//Calculate and Display Total Pay
	for(int i=0;i<size;i++)
	{
		TotalPay+=arr[i].PP*arr[i].PQ;
	}

	cout<< "Total Pay ="<<TotalPay<<endl;
	return 0;
}
```

Home Exercise

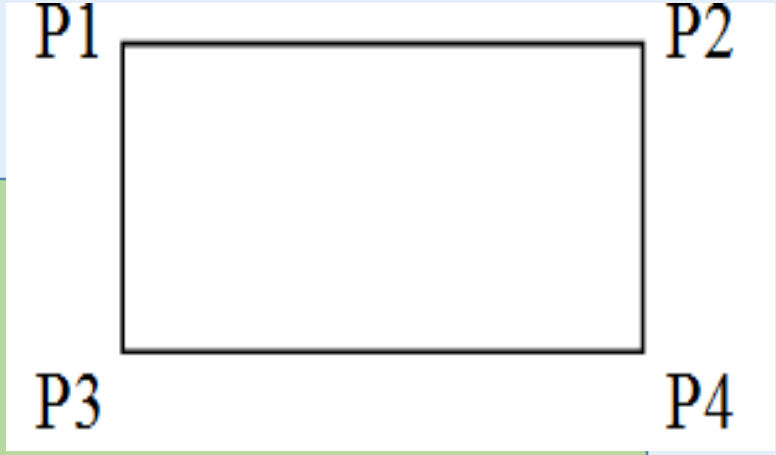
Shape Area with Cartesian Coordinates

Write a program that represents any **point** by a structure including the x-coordinate and the y-coordinate of this point. Then represent each **rectangle** as a structure, including an array of four Cartesian points (which are the four edge points of the rectangle) and the area of this rectangle.



The program asks the user to enter the two **opposite** edge points of a rectangle, calculates the other two edges, and then calculates its area.

The program asks the user to enter P1 and P4, and then calculates P2, P3 and the area.



Sample Execution:

```
Enter Point 1: 3 5
Enter Point 4: 7 2
Point 2 = 7 , 5
Point 3 = 3 , 2
Area is: 12
```

Fractions (Extra Work)

Write a program that represent a **fraction** as structure by its The **numerator** and **denominator** to add two fractions and display the result fraction. Your program will read the two fractions from the user. The numerator and denominator of each fraction are input separately by space.

Sample Execution:

```
Enter fraction 1(numerator denominator): 1 2

Enter fraction 2(numerator denominator): 2 5

Result: 9/10
```

thank you!