

DESCRIPTION

Complete the classes using the Specifications given below. Consider default visibility of classes, data fields, and methods unless mentioned otherwise.

Specifications

```
class definitions:
class Counter:
    data member:
        int[] data
        visibility : public

    Counter(int[] data):Constructor with public visibility

method definition:
    getCount()
        return: String
        visibility: public

class ExceptionZero : Exception:
    method definition:
        ExceptionZero(String message) : base(message)
        visibility: public

class ExceptionOne : Exception:
    method definition:
        ExceptionOne(String message) : base(message)
        visibility: public
```

Task

Class Counter

-define the data members as per the given specifications.

-define the constructor with public visibility.

-the array data will contain 0 or 1 always.

-Implement the below methods for this class:

-String getCount():

- Write a code that gives the output on the below conditions -
 1. If the **count of zeros is even and the count of one is also even** then return "Great".
 2. If the **count of zeros is odd and the count of one is also odd** then return "Great".
 3. If the **count of zeros is even and the count of one is odd** then throw **ExceptionOne** with a message "**One comes odd times**".
 4. If the **count of zeros is odd and the count one is even** then throw **ExceptionZero** with a message "**Zero comes odd times**".

Class ExceptionZero : Exception:

-Define **ExceptionZero** class derived from the Exception class.

Class ExceptionOne : Exception:

-Define **ExceptionOne** class derived from the Exception class.

Sample Input

```
int[] data = {0,1,0,1,1,1,1};
Counter c = new Counter(data);
string res = c.getCount();
Console.WriteLine(res);
```

Sample Output

Great

Instruction

Files in which you have to write the code:

- Counter/Program.cs

Getting to know the IDE

- The button with name **Build** in the IDE builds your project.
- A demo project is already present which will be compiled and deployed by clicking on the **Build** button. Then click on the **Reload Workspace** icon on the project navigation pane.
- -----

Subscribing to a phone call

bookmark_border

- subject Full-stack
- casino 25 points

DESCRIPTION

Subscribing to a phone call:

This problem is related to subscribing to the phone call which will be setting the subscription status accordingly.

Your task here is to implement a **C#** code based on the following specifications. Note that your code should match the specifications in a precise manner. Consider **default visibility** of classes, data fields and methods are public unless mentioned otherwise.

Specifications:

```
class definition:
class PhoneCall:
visibility : default
delegate : Notify()
return type : void
visibility : public
```

```

event : PhoneCallEvent
    type : Notify // delegate type
    visibility : public
property : Message
    return type: string
    get visibility : public
    set visibility : private
    method definitions:
    OnSubscribe()
        return type : void
        visibility : default

    OnUnSubscribe()
        return type : void
        visibility : default

    MakeAPhoneCall(bool notify)
        return type : void
        visibility : public

```

Task:

class PhoneCall:

- **OnSubscribe()**: Sets Message property with string "Subscribed to calls".
- **OnUnsubscribe()**: Sets Message property with string "Unsubscribed to calls".
- **MakeAPhoneCall(bool notify)**: Subscribes to OnSubscribe() when notify is true otherwise subscribes to OnUnSubscribe(). Then it should set the message property accordingly, Also it should delist the subscription for both.

Sample input

```

var call= new PhoneCall();
call.MakeAPhoneCall(true);
Console.WriteLine($"{call.Message}");
call.MakeAPhoneCall(false);
Console.WriteLine($"{call.Message}");

```

Sample output

```

Subscribed to calls
Unsubscribed to calls

```

Instruction

Files in which you have to write the code:

- PhoneCall/program.cs

Getting to know the IDE

- The button with name **Build** in the IDE builds your project.
 - A demo project is already present which will be compiled and deployed by clicking on the **Build** button. Then click on the **Reload Workspace** icon on the project navigation pane.
-

DESCRIPTION

Complete the classes using the Specifications given below. Consider default visibility of classes, data fields, and methods unless mentioned otherwise.

Specifications

```
class definitions:
    class Khata:
        data members:
            Dictionary<string, int> record = new Dictionary<string,
int>();
            visibility : public

            Khata(Dictionary<string, int> record): constructor with
public visibility

        method definitions:
            getTotal():
                return type: int
                visibility: public
            getRepeatAmount():
                return type: int
                visibility: public
```

Task

Class Khata

- define all the variables according to the above specifications.
- define a **constructor** with public visibility.
- The dictionary key implies item name and value implies the amount of that particular item.

Implement the below methods for this class:

-int getTotal():

- Write a code that returns the **sum of the amount that is spent for all the items** on the basis of a given collection **record**.

-int getRepeatAmount():

- Write the code that returns the **count of the unique amount in the given dictionary (record)**.

<https://stackoverflow.com/questions/8459928/how-to-count-occurrences-of-unique-values-in-dictionary>

- **Note - All the keys are unique.**

Sample Input 1

```
Dictionary<string, int> record = new Dictionary<string, int>();
record.Add("milk", 100);
record.Add("tea", 50);
Khata khata = new Khata(record);
khata.getRepeatAmount()
```

Sample Output 1

2

Sample Input 2

```
Dictionary<string, int> record = new Dictionary<string, int>();
record.Add("milk", 100);
record.Add("tea", 50);
Khata khata = new Khata(record);
Console.WriteLine(khata.getTotal());
```

Sample Output 2

150

Instruction

Files in which you have to write the code:

- Khata/Program.cs

Getting to know the IDE

- The button with name **Build** in the IDE builds your project.
- A demo project is already present which will be compiled and deployed by clicking on the **Build** button. Then click on the **Reload Workspace** icon on the project navigation pane.