**Question No : 1**

**1. Prepare Bill**

 Coding

Description

**Prepare Bill:**

This problem is create the bill and find the total amount of the bill.

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.

**Specification:**

class definition:

enum CommodityCategory: Furniture, Grocery, Service

class Commodity:

visibility : default

constructor : Commodity(CommodityCategory category, string commodityName, int commodityQuantity, double commodityPrice)

visibility : public

member property : Category

return type : CommodityCategory

visibility : public

member property : CommodityName

return type : string

visibility : public

member property : CommodityQuantity

return type : int

visibility : public

member property : CommodityPrice

return type : double

visibility : public

class PrepareBill:

visibility : default

member variable : \_taxRates

type : readonly

return type : IDictionary<CommodityCategory, double>

visibility : private

constructor : PrepareBill()

visibility : public

member definition : SetTaxRates(CommodityCategory category, double taxRate)

return type : void

visibility : public

member definition : CalculateBillAmount(IList<Commodity> items)

visibility : public

return type : double

**Task:**

**enum CommodityCategory:**

* Define the enum as mentioned in specification.

**class Commodity:**

* Define the class as mentioned in specification.

**class PrepareBill:**

* **PrepareBill():** Initializes \_taxRates with empty dictionary.
* **SetTaxRates(CommodityCategory category, double taxRate):** Accepts commodity category and tax of that category and then its adds to \_taxRates. If the tax is already added to CommodityCategory then it should not add to \_taxRates.
* **CalculateBillAmount(IList<Commodity> items):** It accepts a list of commodities and it should return the total amount of the bill. If the commodity category is not present in the \_taxRates, then it should throw ArgumentException.

**Sample input:**

var commodities = new List<Commodity>()

      {

        new Commodity(CommodityCategory.Furniture, "Bed", 2, 50000),

        new Commodity(CommodityCategory.Grocery, "Flour", 5, 80),

        new Commodity(CommodityCategory.Service, "Insurance", 8, 8500)

      };

      var prepareBill = new PrepareBill();

      prepareBill.SetTaxRates(CommodityCategory.Furniture, 18);

      prepareBill.SetTaxRates(CommodityCategory.Grocery, 5);

      prepareBill.SetTaxRates(CommodityCategory.Service, 12);

      var billAmount = prepareBill.CalculateBillAmount(commodities);

      Console.WriteLine($"{billAmount}");

**Sample output:**

194580

**Note:**

* You can make suitable function calls and use **the RUN CODE** button to check your **main()** method output.

##### Execution time limit

2 seconds

**Using System;**

**enum CommodityCategory**

**{**

**// your code goes here**

**}**

**class Commodity**

**{**

**// your code goes here**

**}**

**class PrepareBill**

**{**

**// your code goes here**

**}**

**class Source {**

**static void Main(string[] args) {**

**/\* Enter your code here. Read input from STDIN. Print output to STDOUT \*/**

**}**

**}**

**Question No : 2**

**2. Broadband plans**

 Coding

Description

**Broadband Plans:**

This problem is related to implementing the functionality of getting the broadband subscription plan based as per the criteria mentioned below.

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:

interface IBroadbandPlan :

visibility : default

member definition :

GetBroadbandPlanAmount();

return type : int

visibility : default

﻿

﻿class Black : IBroadbandPlan

visibility : default

﻿

member field:

\_isSubscriptionValid

visibility : private

return type : bool

type : readonly

\_discountPercentage

visibility : private

return type : int

type : readonly﻿

PlanAmount = 3000

﻿ visibility : private

return type : int

type : const

﻿

constructor : Black(bool isSusbcriptionValid, int discountPercentage)

visibility : public

﻿

method definition :

GetBroadbandPlanAmount()

visibility : public

return type : int

class Gold: IBroadbandPlan

visibility : default

﻿

member field :

\_isSubscriptionValid

visibility : private

return type : bool

type : readonly

\_discountPercentage

visibility : private

return type : int

﻿ type : readonly

PlanAmount = 1500

visibility : private

return type : int

type : const

constructor : Gold(bool isSusbcriptionValid, int discountPercentage)

visibility : public

member definition :

GetBroadbandPlanAmount()

visibility : public

return type : int

﻿﻿

﻿class SubscribePlan :

﻿ member field :

\_broadbandPlans

visibility : private

return type : IList<IBroadbandPlan>

type : readonly

﻿ constructor : SubscribePlan(IList<IBroadbandPlan> broadbandPlans)

visibility : public

﻿

member definition :

GetSubscriptionPlan()

visibility : public

return type : IList<Tuple<string, int>>

**Task:**

**class Black:**

* **Black(bool isSusbcriptionValid, int discountPercentage):** Accepts two parameters and initializes \_isSubscriptionValid and \_discountPercentage. It should throw ArgumentOutOfRangeException if discountPercentage is less than 0 and greater than 50.
* **int GetBroadbandPlanAmount():** If the subscription is valid then it should return the discounted price otherwise it should return the normal price.

**class Gold:**

* **Gold(bool isSusbcriptionValid, int discountPercentage):** Accepts two parameters and initializes \_isSubscriptionValid and \_discountPercentage. It should throw ArgumentOutOfRangeException if discountPercentage is less than 0 and greater than 30.
* ﻿**int GetBroadbandPlanAmount():** If the subscription is valid then it should return the discounted price otherwise it should return the normal price.

**class SubscribePlan:**

* **SubscribePlan(IList<IBroadbandPlan> broadbandPlans):**Accepts a list of Broadband Plans and initializes the \_customers field. If the input null than it should throw ArgumentNullException.
* **IList<Tuple<string, int>> GetSubscriptionPlan():** It should return a list which contains plan type and its subscription plan amount. If \_broadbandPlans is null than it should throw ArgumentNullException.
* **Sample input:**
* var plans = new List<IBroadbandPlan>
* {
* new Black(true, 50),
* new Black(false, 10),
* new Gold(true, 30),
* new Black(true, 20),
* new Gold(false, 20)
* };
* var subscriptionPlans = new SubscribePlan(plans);
* var result = subscriptionPlans.GetSubscriptionPlan();
* foreach (var item in result)
* {
* Console.WriteLine($"{item.Item1}, {item.Item2}");
* }

**Sample output:**

Black, 1500

Black, 3000

Gold, 1050

Black, 2400

Gold, 1500

**Note:**

* You can make suitable function calls and use **the RUN CODE** button to check your **main()** method output.

##### Execution time limit

2 seconds

**using System;**

**using System.Collections.Generic;**

**interface IBroadbandPlan**

**{**

**// your code goes here**

**}**

**class Black : IBroadbandPlan**

**{**

**// your code goes here**

**}**

**class Gold : IBroadbandPlan**

**{**

**// your code goes here**

**}**

**class SubscribePlan**

**{**

**// your code goes here**

**}**

**class Source {**

**static void Main(string[] args) {**

**/\* Enter your code here. Read input from STDIN. Print output to STDOUT \*/**

**}**

**}**

**Question No 3**

**3. The Internet problem**

 Coding

Description

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

**Specifications**

class definitions:

﻿class Internet:

﻿data member:

﻿ string Name,

int DataLimit,

int Speed

Internet(string Name, int DataLimit, int Speed)﻿: Define a constructor with public visibility

﻿method definition:

﻿ DownloadMovie(int Size):

﻿ ﻿return type: string

﻿ ﻿visibility: public

﻿

﻿ WatchMovie(int Size):

﻿ ﻿return type: string

﻿ ﻿visibility: public﻿

﻿ class InternetException: Exception

method definition:

InternetException(String Message) : base(Message)

visibility: public

﻿

## **Task**

Define a Class **Internet**according to the above specifications and implement the below methods for this class:

**-string DownloadMovie(int Size):**

* throw an InternetExceptionwith the message "**File too large**" if the value of Datalimit is less than Size.
* throw an InternetExceptionwith the message "**Low Bandwidth**" if Speed is less than 200**.**
* throw an InternetExceptionwith the message "**Time exceeded**" if Time taken to download the file is greater than 100(**Assume the speed is given in kbps and the size of file is in MB**)**.**
* **If no exception is thrown then return "Can be downloaded".**

**-string WatchMovie(int Size):**

* If DownloadMovie() returns an exception then return "**Cannot be downloaded"(Use try and catch block).**
* If any other exception is thrown then return "**Other exception**".
* Return "**Watch listed**" if no error is thrown.

Class **InternetException**

* define custom exception class **InternetException**by **extending** the **Exception** class.
* define a parameterized constructor with a String argument to pass the message to the super class.

**Sample Input**

Internet obj = new Internet("The Paycheck",200,100);

obj.DownloadMovie(200);

**Sample Output**

Can be downloaded

## **NOTE:**

* You can make suitable function calls and use **the RUN CODE** button to check your **main()** method output.

##### Execution time limit

5 seconds

**using System;**

**using System.Collections.Generic;**

**using System.IO;**

**using System.Linq;**

**class Internet{**

**//Your Code Goes Here..**

**}**

**class InternetException : Exception{**

**//Your Code Goes Here..**

**}**

**class Source {**

**static void Main(string[] args) {**

**/\* Enter your code here. Read input from STDIN. Print output to STDOUT \*/**

**}**

**}**