Programming II – Test 1 (Prescription)

# You have 90 minutes to complete all the tasks.

Your company was asked to build a contact manager for a pharmaceutical store, the software architects of your company have designed the system and your supervisor has assigned the task of coding two classes. The two classes are a Drug class and a Prescription class both of them are fully described below.

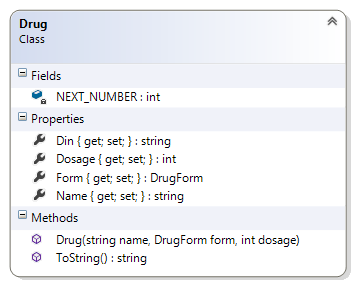
A test harness is provided to test your classes. You are required to match the provided output EXACTLY!

# The DrugForm Enum 5 marks

This enum describes the drug form. The explanation are as follows:

This enum must be defined in a scope so that the other class will be able to access with miminal problems.

# The Drug Class 23 marks

This class is used to capture information on a pharmaceutical Drug.

##### Fields:

**NEXT\_NUMBER** – this private static int represents the value to be used when creating a Drug object. It is initialized to 101. This variable is used and updated in the constructor **public Drug(string name, …)**.

**3 Marks**

**Din** – this string is the drug identity number. This member is set in the constructor. The class variable **NEXT\_NUMBER** is used to generate a unique string. This is a public **readonly** field.

**2 Marks**

##### Properties:

**2 Marks**

**Dosage** – this int represents the dosage of this object. This is an auto-implemented property, the getter is public and the setter is private.

**Form** – this represents the DrugForm of this Drug object. This is an auto-implemented property, the getter is public and the setter is private.

**2 Marks**

**Name** – this double represents the amount that is owed by this Customer. This is an auto-implemented property, the getter is public and the setter is private.

**2 Marks**

##### Constructor:

**public Drug(string name, int dosage, DrugForm form = DrugForm Tablet)**  – This is constructor does the following:

**7 Marks**

* Assigns the arguments to the appropriate properties.
* It also assigns the **NEXT\_NUMBER** field to the **DIN** property (you will have to do some kind of conversion) and increments it.
* The last argument has a default value.

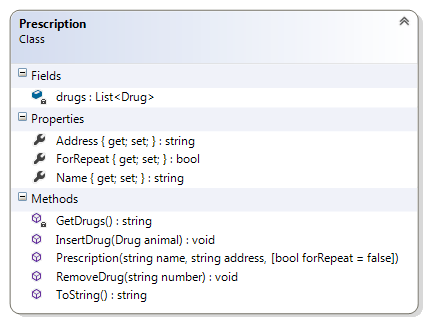
##### Methods

**public override string ToString()** – This method overrides the corresponding method of the object class to return a suitably formatted string. See the sample output for ideas on how to format your output.

**5 Marks**

This method does not display anything.

# The Prescription Class 40 Marks

We are going to model a Prescription type. There are 8 members in this class as shown in the class diagram below.

#### Description of class members

##### Fields:

**3 Marks**

**drugs** – this is a list of Drugs. It represents a collection of Drugs that comprise this Prescription. This is initialized at declaration. This field is private.

##### Properties:

**2 Marks**

**Name** – this string represents the name of the person for this prescription. This is an auto-implemented property, the getter is public and the setter is private.

**ForRepeat** – this bool indicates if this prescription is to be repeated. This is an auto-implemented property, the getter is public and the setter is private.

**2 Marks**

**Address** – this string represents the address of the patient. This is an auto-implemented property, the getter is public and the setter is private.

**2 Marks**

##### Constructor:

**public Prescription(string name, string address, bool forRepeat = false)** – This is constructor assigns the arguments to the appropriate properties. Note the last argument has a default value.

**3 Marks**

##### Methods

**public void InsertDrug(Drug drug)** – This public method add the argument to the field **drugs**.

**3 Marks**

This method does not display anything.

**public void RemoveDrug(string din)** – This public method removes a Drug from the collection of Drugs. This method uses an appropriate loop to check each Drug in the collection. If the **Din** property of that Drug matches the argument then that particular Drug is removed from the collection. If the Drug could not be found then an **Exception** object with a suitable message is thrown. [Use the method **RemoveAt(i)** of the list class to delete the Drug from the collection].

**10 Marks**

You should not use a **foreach** loop in this method, because it iterates in a readonly fashion so you will not be able to remove it.

Use either a **for** or a **while** or a **do-while** loop

This method does not display anything.

**private string GetDrugs()** – This is a private method that returns a string representing all the elements of the drugs collection. There is a single line for each element. This method is used in the **ToString()** method below to print a Store. [To get a new line use the "\n" sequence].

**8 Marks**

This is method does not display anything.

**public override string ToString()** – This is a public method overrides the corresponding method in the object class to return a stringified form of the object. In addition to the **Name**, **Address** and **ForRepeat** properties, this method uses the **GetDrugs()** method to generate a string for all the Drugs. Examine the output to decide on your formatting code.

**6 Marks**

This method does not display anything.

### Test Harness

Insert the following code statements in your Program.cs file:

//test the Drug class

Console.WriteLine("\n\*\*\*\*\*Testing the Drug Class");

Console.WriteLine(new Drug("Aspirin", 85));

Console.WriteLine(new Drug("Tylenol", 125, DrugForm.Capsule));

Console.WriteLine(new Drug("Metformin", 250));

//test the Prescription class

Console.WriteLine("\n\*\*\*\*\*Testing the Prescription Class");

Console.WriteLine(new Prescription("Joanne Fillotti", "Markham Road"));

//testing InsertDrug method of the Prescription class

Console.WriteLine("\n\*\*\*\*\*Testing the InsertDrug()");

Prescription store0 = new Prescription("Jake Nesovich", "Morningside Avenue", false);

store0.InsertDrug(new Drug("Oxycontin", 150, DrugForm.Gel));

store0.InsertDrug(new Drug("Marjuana", 200, DrugForm.Paste));

store0.InsertDrug(new Drug("Amoxicillin", 350, DrugForm.Capsule));

store0.InsertDrug(new Drug("Fentanyl", 50, DrugForm.Aerosol));

Console.WriteLine(store0);

Console.WriteLine("\n\*\*\*\*\*Testing the InsertDrug()");

Prescription store1 = new Prescription("Bindu Ggoel", "Williams Parkway", true);

store1.InsertDrug(new Drug("Warfarin", 125, DrugForm.Gel));

store1.InsertDrug(new Drug("Prozac", 300, DrugForm.Paste));

store1.InsertDrug(new Drug("Ibuprofen", 250));

store1.InsertDrug(new Drug("Oxycodone", 85, DrugForm.Liquid));

store1.InsertDrug(new Drug("Ropinirole", 125, DrugForm.Cream));

store1.InsertDrug(new Drug("Tramadol", 250, DrugForm.Powder));

Console.WriteLine(store1);

//testing the RemoveCustomer method of the invient class

//check the previous display to verify that atleast

//two of the item numbers are used below

Console.WriteLine("\n\*\*\*\*\*Testing the RemoveDrug()");

store1.RemoveDrug("109");

store1.RemoveDrug("110");

try

{

store1.RemoveDrug("109");

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

Console.WriteLine(store1);

### Sample Output

The following the output of a completed solution. Examine the output carefully to decide on the return value of the **ToString()** of the Drug class and the **ToString()** method of the Prescription class.

\*\*\*\*\*Testing the Drug Class

101 Aspirin 85g (Tablet)

102 Tylenol 125g (Capsule)

103 Metformin 250g (Tablet)

\*\*\*\*\*Testing the Prescription Class

Joanne Fillotti, Markham Road

List of drugs:

\*\*\*\*\*Testing the InsertDrug()

Jake Nesovich, Morningside Avenue

List of drugs:

104 Oxycontin 150g (Gel)

105 Marjuana 200g (Paste)

106 Amoxicillin 350g (Capsule)

107 Fentanyl 50g (Aerosol)

\*\*\*\*\*Testing the InsertDrug()

Bindu Ggoel, Williams Parkway(R)

List of drugs:

108 Warfarin 125g (Gel)

109 Prozac 300g (Paste)

110 Ibuprofen 250g (Tablet)

111 Oxycodone 85g (Liquid)

112 Ropinirole 125g (Cream)

113 Tramadol 250g (Powder)

\*\*\*\*\*Testing the RemoveDrug()

Error: drug 109 not found

Bindu Ggoel, Williams Parkway(R)

List of drugs:

108 Warfarin 125g (Gel)

111 Oxycodone 85g (Liquid)

112 Ropinirole 125g (Cream)

113 Tramadol 250g (Powder)

|  |
| --- |
| [Solution]  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Prescription  {  class Program  {  // 4. Test Harness  static void Main(string[] args)  {  //test the Drug class  Console.WriteLine("\n\*\*\*\*\*Testing the Drug Class");  Console.WriteLine(new Drug("Aspirin", 85));  Console.WriteLine(new Drug("Tylenol", 125, DrugForm.Capsule));  Console.WriteLine(new Drug("Metformin", 250));  //test the Prescription class  Console.WriteLine("\n\*\*\*\*\*Testing the Prescription Class");  Console.WriteLine(new Prescription("Joanne Fillotti", "Markham Road"));  //testing InsertDrug method of the Prescription class  Console.WriteLine("\n\*\*\*\*\*Testing the InsertDrug()");  Prescription store0 = new Prescription("Jake Nesovich", "Morningside Avenue", false);  store0.InsertDrug(new Drug("Oxycontin", 150, DrugForm.Gel));  store0.InsertDrug(new Drug("Marjuana", 200, DrugForm.Paste));  store0.InsertDrug(new Drug("Amoxicillin", 350, DrugForm.Capsule));  store0.InsertDrug(new Drug("Fentanyl", 50, DrugForm.Aerosol));  Console.WriteLine(store0);  Console.WriteLine("\n\*\*\*\*\*Testing the InsertDrug()");  Prescription store1 = new Prescription("Bindu Ggoel", "Williams Parkway", true);  store1.InsertDrug(new Drug("Warfarin", 125, DrugForm.Gel));  store1.InsertDrug(new Drug("Prozac", 300, DrugForm.Paste));  store1.InsertDrug(new Drug("Ibuprofen", 250));  store1.InsertDrug(new Drug("Oxycodone", 85, DrugForm.Liquid));  store1.InsertDrug(new Drug("Ropinirole", 125, DrugForm.Cream));  store1.InsertDrug(new Drug("Tramadol", 250, DrugForm.Powder));  Console.WriteLine(store1);  //testing the RemoveCustomer method of the invient class  //check the previous display to verify that atleast  //two of the item numbers are used below  Console.WriteLine("\n\*\*\*\*\*Testing the RemoveDrug()");  store1.RemoveDrug("109");  store1.RemoveDrug("110");  try  {  store1.RemoveDrug("109");  }  catch (Exception e)  {  Console.WriteLine(e.Message);  }  Console.WriteLine(store1);  }  }  }  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Prescription  {  // 1. DrugForm enum: 5 marks  enum DrugForm  {  Liquid,  Cream,  Gel,  Paste,  Tablet,  Capsule,  Powder,  Aerosol  }  }  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Prescription  {  // 2. Drug Class: 23 Marks  class Drug  {  // FIELDS  private static int NEXT\_NUMBER = 101; // 3 Marks  public readonly string Din; // 2 Marks  // PROPERTIES  public int Dosage { get; private set; } // 2 Marks  public DrugForm Form { get; private set; } // 2 Marks  public string Name { get; private set; } // 2 Marks  // CONSTRUCTOR  public Drug(string name, int dosage, DrugForm form = DrugForm.Tablet) // 7 Marks \*  {  Name = name;  Dosage = dosage;  Form = form;  Din = Convert.ToString(NEXT\_NUMBER);  NEXT\_NUMBER++;  }  // METHODS  public override string ToString() // 5 Marks  {  return string.Format("{0} {1} {2}g ({3})", Din, Name, Dosage, Form);  }  }  }  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Prescription  {  // 3. Prescription class: 40 Marks  class Prescription  {  // FIELDS  private List<Drug> drugs = new List<Drug>() { }; // 3 Marks  // PROPERTIES  public string Name { get; private set; } // 2 Marks  public bool ForRepeat { get; private set; } // 2 Marks  public string Address { get; private set; } // 2 Marks  // CONSTRUCTOR  public Prescription(string name, string address, bool forRepeat = false) // 3 Marks  {  Name = name;  Address = address;  ForRepeat = forRepeat;  }  // METHODS  public void InsertDrug(Drug drug) // 3 Marks \*  {  drugs.Add(drug);  }  public void RemoveDrug(string din) // 10 Marks \*\* - 5  {  bool checkDin = true;  for (int i = 0; i < drugs.Count; i++)  {  if (drugs[i].Din.Contains(din))  {  drugs.RemoveAt(i);  checkDin = false;  }  }  if(checkDin)  throw new Exception("Error: drug " + din + " not be found");  }  private string GetDrugs() // 8 Marks \* - 4  {  string drugList = "";  foreach (Drug x in drugs)  {  drugList += " " + x + "\n";  }  return drugList;  }  public override string ToString() // 6 Marks \*  {  return String.Format(Name + ", " + Address + "\nList of drugs: \n" + GetDrugs());  }  }  } |