Lab Guide
Hands-On Manual- C#
V-0.1

C# Generate Product Code

Problem Statement: Generate Product Code

Create a class named Product with the following **private** instance variables
- productCode of type String
- productName of type String
- productPrice of type double
- categoryCode of type char (E- electronics, A - apparels, T – toys)

Include a private static variable prodCounter of type int initialized to 100.

Create public getters and setters for all variables.

Create a private method generateProductCode which will return the product code as String. Product code is derived by concatenating categoryCode and incremented product counter.

Include a parameterized constructor with 3 parameters (productName, productPrice, categoryCode). productCode should be assigned in constructor by using generateProductCode method. Initialize all the member variables.

Include an overloaded parameterized constructor with 2 parameter(productName, productPrice). productCode should be assigned in constructor by using generateProductCode method. Category code should be assigned to 'E'. Initialize all the member variables.

Include a method getProductDetails to format the product details. This method should return a String containing the product details in the below format.

Code–101E, Name – Laptop, Price - 45000.00, Category – E

Assuming that the above product object is constructed with the help of below statement
Product p=new Paroduct("Laptop",45000.00, 'E');

Note : Do not implement the Main Method

Binary Tree Serialization V2

Problem Statement: Binary Tree Serialization

Serialization is to store a tree in a file so that it can be later restored. The structure of tree must be maintained. **Deserialization** is reading the tree back from the file.

Task to implement:

- Implement the insert function of BST.
- Produce a pre-order traversal of the intermediate file generated from the **Serialised file**. This field's contents should be printed to **STDOUT** as the output.

Note: For more info check the images below.

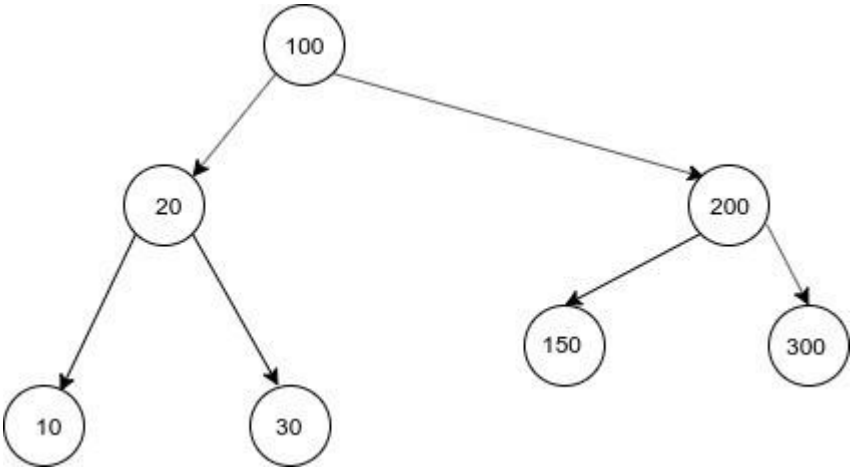

Fig. 1 Serialized File

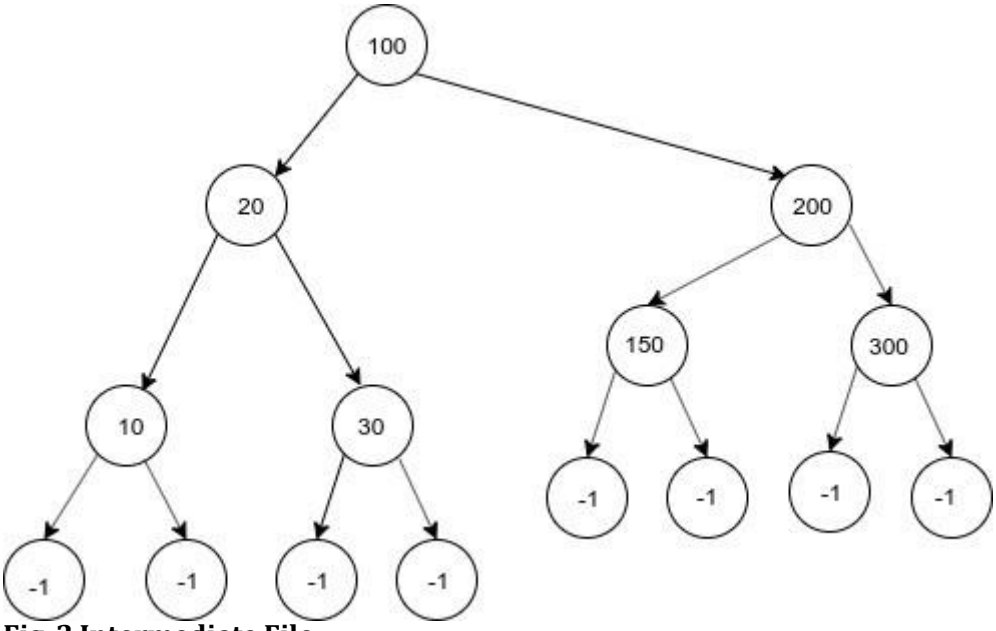

Fig. 2 Intermediate File

NOTE

- Main function has already been taken care of in the boilerplate, don't change it or you may not get the expected output.
- Make sure you strictly follow the given **Output Format**, else your test-cases will fail.

Input Format

- The first line contains the **number of nodes in the BST** which is already taken in Main(), represented by **N**
- The next **N** lines contains the BST elements, where:
- the first element is the root node.
- the subsequent elements to be decided based on BST conventions.

Output Format

- This is the pre-order representation of the intermediate file.
- For every leaf node, the left and the right children are represented as **-1**.

Sample Test Cases -
Sample Input

```
7
100
20
10
30
200
150
300
```

```
100
20
10
-1
-1
30
-1
-1
200
150
-1
-1
300
-1
-1
```

Address Book V2

Problem Statement: Address Book
The 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 unless mentioned otherwise.

Specifications

```
class PersonAddress:
 method definition:
  Type: Implement getter setter method.
  return type: string
  visibility: public

  HouseNo: Implement getter setter method.
  return type: string
  visibility: public

  StreetName: Implement getter setter method.
  return tye: string
  visibility: public

  City: Implement getter setter method.
  return type: string
  visibility: public

class PersonalDetails:
 method definition:
  Name : Implement getter setter method.
  return type: string
  visibility: public

  Age: Implement getter setter method.
  return type: int
  visibility: public

  Company: Implement getter setter method.
```

```
  return type: string
  visibility: public

  PAN: Implement getter setter method.
  return type: string
  visibility: public

  AddressList: Implement getter setter method.
  return type: List<PersonAddress>
  visibility: public

class Program:
 method definition:
  Serialize: serialize List of PersonalDetails object to xml file (xmloutput.xml)
  return type: string
  return message Successful if serialization is successful.

  DeSerialize: Deserializae xml file (xmloutput.xml) into List of PersonalDetails object
  return type: List<PersonalDetails>
```

Task to implement:

- Create classes **PersonAddress** and **PersonalDetails** according to above specifications.
- Create a class **Program** and implement the below-given methods:

1. **Serialize:** serialize list to XML file (xmloutput.xml)
2. **Deserialize:** deserialize the XML file into a list collection of PersonalDetails object.

- Use the XMLElement attribute to define Order of XML tags.
- Don't add PAN property to xml file while serializing
- Use XMLAttribute for HouseNo from PersonAddress class to add as XML attribute (don't create xml tag for HouseNo)
- Serialize List of PersonalDetails object to 1 XML file.

IMPORTANT

If you want to test your program you can implement a Main() function and you can use RUN CODE to test your Main() provided you have made valid function calls with valid data required.

Account Details

Problem Statement - Account Details

Complete the class Account and AccountDetails as per the below requirement.
class Account:
Create the following instance/static members: - accountNo : int - balance : double - accountType : String - counter :int static
Define parameterized constructor with two parameters to initialize balance and accountType. accountNo should be initialized by incrementing counter.

Task to implement:

- Implement the below operations:
- void depositAmount(double amount)
- To add amount to account balance
- void printAccountDetails()
- To display account details as per format given in Example Section

class AccountDetails :

- Create main method and follow the below instructions.
- Accept balance, account type and amount as input for two account objects from Console(Refer Example section for input format)
- create first object using the input data and display account details.
- Deposit amount using the input data and display the new account balance.
- create second account object using the input data and display account details.
- Set account balance to new balance using input data and display the new account balance

Example

```
Sample Input:
100.5
Savings
25.5                                    // balance type amount for first account
200
Current
50.5                                    // balance type amount for second account

Expected Output:
[Acct No : 1, Type : Savings, Balance : 100.5]
New Balance : 126.0
[Acct No : 2, Type : Current, Balance : 200.0]
New Balance : 50.5
Sample Input:
0
```

```
Current
100
0
Current
50

Expected Output:
[Acct No : 1, Type : Current, Balance : 0.0]
New Balance : 100.0
[Acct No : 2, Type : Current, Balance : 0.0]
New Balance : 50.0
```

Instructions

- Do not change the provided class/method names unless instructed.
- Ensure your code compiles without any errors/warning/deprecations.
- Follow best practices while coding.
- Avoid too many & unnecessary usage of white spaces (newline, spaces, tabs, ...), except to make the code readable
- Use appropriate comments at appropriate places in your exercise, to explain the logic, rational, and solutions, so that the evaluator can know them
- Try to retain the original code given in the exercise, to avoid any issues in compiling & running your programs.
- Always test the program thoroughly, before saving/submitting exercises/project
- For any issues with your exercise, contact your coach.

Warnings

- Take care of whitespace/trailing whitespace
- Trim the output and avoid special characters.
- Avoid printing unnecessary values other than expected/asked output.

Valid List check_circ

Henry has to design a coding problem for a placement drive. But the problem is that he is not good at coding rather he is a good problem analyzer. Analyze the problem and write a code for the same.

Problem Statement: Valid List Check_circ

You are given a list of natural numbers and a number k.
A valid list has the following properties. –

1. It only contains value < k.
2. It only contains values whose adjacent digit difference is 0, 1, or -1.

Input format

- First line contain the value of N, the number of queries.
- For each query -

1. First line contain space separated value of L (size of array) and K.
2. Second line contain space separated values of array.

Output Format

- For each query print the space separated list in a new line.

Constraints

- 1 <= **N** <= 1000
- 1 <= **length of array** <= 10000
- 1 <= **A[i]** <= 10000

Sample Input

```
2
5 20
23 12 8 7 11
4 45
12 22 55 8
```

Sample Output

```
12 11
12 22
```

Explanation

For query 1 -
Step 1 - Taking number less than k = 20. Updated array = [12, 8, 7, 11]

Step 2 - Eliminating number whose adjacent digit difference is not equal to 1, 0, -1.

- for element 1 -> 1-2 = -1
- for element 2 -> 8
- for element 3 -> 7
- for element 4 -> 1-1 = 0

Therefore, Valid list -> [12,11]

Evaluation details

Evaluation mode

I/O testcase based

Score

100

Blackbox testing

Serializing Book

Problem Statement: Serializing Book

The 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 unless mentioned otherwise.

Specifications:

```
class definitions:
  class Book:
     method definitons:
       Name: Implement getter setter method (use Auto Implementation Property)
          return type: string
          visibility: public
       Price: Implement getter setter method (use Auto Implementation Property)
          return type: string
          visibility: public
       Author: Implement getter setter method (use Auto Implementation Property)
          return type: string
          visibility: public
       Year: Implement getter setter method (use Auto Implementation Property)
          return type: string
          visibility: public
       Book(string name, string price, string author, string year) : constructor
          visibility: public
       Ser(List<Book> books) : method to implement serialization
          return type: stream
          visibility: public
          return: stream(serialized list in binary format)
       Deser(FileStream s): method to implement deserialization
          return type: List
          visibility: public
          return: deserialized list
       main(String args[]): method of type static void
             List<Book> list: List
             s:  FileStream
             method calls:
                Ser(list)
```

Deser(s)

Task to implement:

Create a Book class with **string Name, string Price, string Author, string Year attributes,** your task is to implement the below given methods to perform serialization and deserialization.

- Define getter setter method using **Auto Implementation Property**
- Define parameterized constructor.
- Implement **Ser(List<Book> books)** method to serialize List<Book>. The serialization, which takes place should be done by sending the Serialize message to the BinaryFormatter object and serialize it to the file called bks.txt.(The serialization relies on a binary stream, represented by an instance of class Filestream)
- Implement **Deser(FileStream s)** method to deserialize the list from the file .

Note:
The class which needs to be serialized needs to have the [Serializable] attribute.
IMPORTANT:

- If you want to test your program you can implement a Main() function given in the stub and you can use RUN CODE to test your Main() provided you have made valid function calls with valid data required.

Sample Input

```
Book first = new Book( "Alchemist", 175, "Paulo Coelho", 1988 );
```

Palindrome Destroyer

Problem Statement: Palindrome Destroyer

John asked for a puzzle from one of his friends. He has been given a string and he has to decode the given string according to the set of rules –

1. Reverse each word of the space-separated string.
2. Eliminate palindrome words.

Palindrome words are those words that can be read the same from either side. For example – "aba" is the same as the reverse of "aba"; Therefore, it is a palindrome.

Input Format

- First line contains the value of **N**, no. of queries(String).
- Next **N** lines contains string.

Output Format

- For each query print the decoded string in a newline.

Constraints

- 1 <= **N** <= 100
- 1 <= **length of s** <= 1000

Sample Input

```
2
i love my country
she is madam
```

Sample Output

```
evol ym yrtnuoc
ehs si
```

Explanation
Query 1 -

- After reversing each word - **I evol ym yrtnuoc**.
- After eliminating palindrome words - **evol ym yrtnuoc**

Query 2 -

- After reversing each word - **ehs is madam**
- After eliminating palindrome words - **ehs is**

- Evaluation details
- **Evaluation mode**
- I/O testcase based
- **Score**
- 50

BlackBox Testing

Measure the Area

Problem Statement: Measure the Area

Jake's school teacher gave him the assignment to write a C# program that calculates the area of a convex quadrilateral. The quadrilateral is described by the coordinates of four 2-dimensional points: *(x1, y1), (x2, y2), (x3, y3)* and *(x4, y4)*. Jake is busy doing his Mathematics assignment. So, he asks for your help to complete the C# assignment for him, since you are good at it. He remembers that he has already done a similar assignment for the triangle. So, he suggests you take help of it.
The classes he had created already are described below. Consider default visibility of classes, data fields, and methods unless mentioned otherwise.

```
class Point:
 data fields:
  x: integer variable with public accessibility denoting the x co-ordinate of the point
 y: integer variable with public accessibility denoting the y co-ordinate of the point
 methods:
  Point:
   Constructor with public accessibility to initialise the point

class Triangle:
 data fields:
   p1, p2, p3: Three Point objects with public accessibility denoting the
               points that describes the triangle.
 methods:
   Triangle:
     Constructor with public accessibility responsible to initialise the triangle
   getArea:
     A method with public accessibility which returns a double variable denoting the area
     of the triangle
```

Task to implement:

create a class named ***Quadrilateral*** which should be a subclass of ***Triangle***. The description is given below:

```
class Quadrilateral extending the class Triangle:
 data fields:
   p4: The fourth point of the quadrilateral with public accessibility
```

methods:
Quadrilateral:
Constructor with public accessibility responsible **to** initialize the quadrilateral
getArea: //overridden method
Returns a **double variable** denoting the area **of** the quadrilateral, **use** the getArea method **of** Triangle **class to** calculate it

Constraints

- All the coordinates lie between **-100** and **100**.
- The points are given in either clockwise or anti-clockwise order.

You don't need to write the main function.

Input

An object **of** Triangle **class or Quadrilateral class**.
(You don't need to process any input)

Output

A double variable denoting the area **of** the Triangle **or** the Quadrilateral. (You don't need to output anything)