System Requirements Specification Index

For

C# .Net Core 3.1 Skills Evaluation

Version 4.0

IIHT Pvt. Ltd.
IIHT Ltd, No: 15, 2nd Floor, Sri Lakshmi Complex, Off MG Road, Near SBI LHO,
Bangalore, Karnataka – 560001, India
fullstack@iiht.com

.NET CORE SKILLS EVALUATION

System Requirements Specification

1. BUSINESS-REQUIREMENT:

1.1 USE CASE DESCRIPTION

Use Case 1

1 Write method of date class to support the following: -

a) Method for validating that the integer-representing month is between 1 & 12 and checking that the day part of the date objects is within the correct range for a month.

b) Obtaining the next day from a given date.

Description

1. Take console input of a date, in **Main ()** method, from user
2. Pass the dateTime to method '**bool CheckMonthRange (string dateTime)**' and write the logic to check if the MonthPart of date is in the range (between 1-12) or not.
   Returns the result to Main method and display the result
   Return 'true' – if Month is in the range
   Return 'false' – if Month is not in the range
3. Pass the dateTime to method '**bool CheckDayRange (string dateTime)'** and write the logic to check the DatePart of date is in the range (between 1-31/1-30/1-28/1-29) or not.
   Return the result to the Main method and display the result.
   Return 'true' – if Datepart is in the range
   Return 'false' – if Datepart is not in the range
   (Subfunction **int GetDaysOfMonth (string dateTime)** can be used to get the TOTAL number of days in the provided month ().
   For the months: `1, 3, 5, 7, 8, 10, 12 – will give 31 days`
   `4, 6, 9, 11 - will give 30 days`
   `2 - will give 29(if it is leap year), else 28`
4. Pass the dateTime to method '**String GetNextDay (string dateTime)',** and write the logic to get the Next Day of the provided date.
   Return the result to the Main method and display the result.

Use Case 2

Take input of two numbers from users and find out HCF of those numbers.

Description

1. Take console input of two numbers, in **Main ()** method from user.
2. Pass the string to method '**int FindHCF (int n1, int n2)'** and write the logic in that method to find HCF of two numbers:

Return - an integer result

3. Display the result.

Use Case 3

Take input of a number from the user and convert it into words and store them in a string.

Description

1. Take console input of a string, in the **Main ()** method from the user.
2. Pass the string to method '**string WriteIntToString (int number)'** and write the logic in that method to convert the number to string:
   Return – result string
3. Display the result.

2. Considerations

- Your code will also be evaluated for code quality, naming conventions, readability etc.
- Make sure you do not modify existing class and method names and their signatures, else it would severely affect the final result.
- Make sure you do not add any new class or methods, else it would severely affect the final result.
- Make sure you do not modify any test cases, else it would severely affect final result

3.Resources available:

3.1 Package: DateClass

Names	Resource	Remarks	Status
Package Structure/Project			
HCF	Program.cs	This file contains business logic to return HCF.	Partially Implemented
DateClass	Program.cs	This class contains all the business logic related to days.	Partially Implemented
IntToString	Program.cs	This class contains business logic to convert a number from int to string.	Partially Implemented

3.2 Package: YakshaEvaluation_Test

Resources

Note: - Under the YakshaEvaluation_Test contains all test cases for code evaluation, please don't try to alter and edit it.

4. Execution Steps to Follow

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**

2. **To open the command terminal the test takers need to go to the Application menu (Three horizontal lines at left top) Terminal → New Terminal.**

3. **You need to follow steps (4,5,6,8) for all 3 use cases.**
   Project folders: (DateClass, HCF, IntToString)

4. **On command prompt, cd into your project folder (cd <Your-Project-folder>).**

5. **To build your project use command:**
   (<Your-Project-folder>/ dotnet build)

6. **To launch your application, Run the following command to run the application:**
   (<Your-Project-folder> / dotnet run)

7. **This editor Auto Saves the code.**

8. **To run the test cases in CMD, Run the following command to test the application:**
   (YakshaEvaluation_Test/dotnet test --logger "console;verbosity=detailed")
   (You can run this command multiple times to identify the test case status,
   and refactor code to make maximum test cases passed before final submission)

9. **If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B - command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.**

10. **These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.**

11. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**