# Review Test Submission: Online Midterm

| | |
|---|---|
| User | Rahul Devarajan Raj |
| Course | Introduction to Programming |
| Test | Online Midterm |
| Started | 6/28/21 3:34 PM |
| Submitted | 6/28/21 4:35 PM |
| Status | Needs Grading |
| Attempt Score | Grade not available. |
| Time Elapsed | 1 hour, 1 minute out of 2 hours |

Instructions

- **You can only attempt the Midterm Exam once. Once you start, you have 120 minutes to complete.**
- **It is an open-book exam, but you cannot search over the Internet. Any use (a part or whole) of online materials found from the Internet will be considered as cheating. Suspicious cheating cases will be reported to the College;**
- **You can only use the files, if any, provided in the question. You are not allowed to change the filename unless instructed. Otherwise, zero marks will be awarded to the whole exam.**
- **You are not allowed to share this paper or your work, in any form, to others. Violation will be reported to College.**
- **Save files, if needed, frequently when working with your PC;**
- **Answers will be automatically submitted at the end of the exam; Unsaved and un-uploaded files may be lost;**
- **No (re-)submission will be accepted after the Exam;**
- **Instructors have no responsibility for any network interruption, PC failure, file corruption, etc.**

| | |
|---|---|
| Results Displayed | Submitted Answers |

## Question 1

**File to submit:** `Q1.cs`

**Question:**

You are asked to implement an application called **`Q1.cs`**. This application contains two methods in addition to the **`Main`** method. These two methods are as follows:

- **`PrintSolidTriangle`** is a method without return value. It has one integer parameter. This method uses '**`#`**' to print an isosceles triangle according to the parameter. [*See Sample Runs #1-4*]
- **`PrintEmptyTriangle`** is a method without return value. It has one integer parameter. This method uses '**`#`**' to print an isosceles triangle according to the parameter, but the body of the triangle is empty (or filled with space). [*See Sample Run #5-8*]

The **`Main`** method will do the followings:

a. It prompts and asks the user to choose between printing solid or empty triangle;
b. It prompts and asks the user to input the size of the triangle;
c. It prints the corresponding triangle accordingly.

**Additional Requirement and Assumption:**

1. **Zero** marks will be given if the program is not compilable;
2. **Zero** marks will be given if you do not following the naming of methods/parameters stated above;
3. You are NOT allowed to use any pre-defined classes except System.Console;
4. You are NOT allowed to use array or any collection classes;
5. You are NOT allowed to use any methods related to the string type (e.g., Replace, Remove, string concatenation, etc.);
6. You may assume the input is valid when testing the program (e.g, the size of the triangle must be greater than or equal to 1).

**Sample Run #1 (Green text refers to user input):**

```
Please choose:
1) Solid Triangle
2) Empty Triangle
1
Please enter the size >> 5
    #
   ###
  #####
 #######
#########
```

**Sample Run #2:**

```
Please choose:
1) Solid Triangle
2) Empty Triangle
1
Please enter the size >> 1
#
```

**Sample Run #3:**

```
Please choose:
1) Solid Triangle
2) Empty Triangle
1
Please enter the size >> 2
 #
###
```

**Sample Run #4:**

```
Please choose:
1) Solid Triangle
2) Empty Triangle
1
```

```
Please enter the size >> 3
  #
 ###
#####
```

**Sample Run #5:**

```
Please choose:
1) Solid Triangle
2) Empty Triangle
2
Please enter the size >> 5
    #
   # #
  #   #
 #     #
#########
```

**Sample Run #2:**

```
Please choose:
1) Solid Triangle
2) Empty Triangle
2
Please enter the size >> 1
#
```

**Sample Run #3:**

```
Please choose:
1) Solid Triangle
2) Empty Triangle
2
Please enter the size >> 2
 #
###
```

**Sample Run #4:**

```
Please choose:
1) Solid Triangle
2) Empty Triangle
2
Please enter the size >> 3
  #
 # #
#####
```

Selected Answer: Q1.cs

**Question 2**

**File to submit:** Q2.cs

**Question:**

You are asked to implement an application called **`Q2.cs`**. This application contains one method, **`ValidateUPC`**, in addition to the **`Main`** method. The details of this method are as follows:

- **ValidateUPC** is a boolean method;
- It contains one string parameter, representing one Universal Product Code (UPC), which contains 12 digits;
- The method returns true if the UPC is valid, and false otherwise;
- When calling this method, the method validates the UPC as follows:
    1. The last digit (the rightmost digit) is called the check digit.
    2. Add the digits in the odd-numbered positions from the right (first, third, fifth, etc. - not including the check digit) together and multiply by three.
    3. Add the digits (up to but not including the check digit) in the even-numbered positions (second, fourth, sixth, etc.) to the result.
    4. Take the remainder of the result divided by 10. If the remainder is equal to 0 then use 0 as the check digit, and if not 0 subtract the remainder from 10 to derive the check digit.
        1. Example: Let 036000241457 be the UPC.

| Position | Digit in UPC (a) | Odd Position / Even Position / Check Digit |
|---|---|---|
| 1 | 0 | Odd |
| 2 | 3 | Even |
| 3 | 6 | Odd |
| 4 | 0 | Even |
| 5 | 0 | Odd |
| 6 | 0 | Even |
| 7 | 2 | Odd |
| 8 | 4 | Even |
| 9 | 1 | Odd |
| 10 | 4 | Even |
| 11 | 5 | Odd |
| 12 | 7 | Check Digit |

a) Add all odd position digits together and multiply it by 3: (0 + 6 + 0 + 0 + 2 + 1 + 5) X 3 = 52;

b) Add all even position digits (excluding check digit) to the result: 52 + 3 + 0 + 0 + 4 + 4 = 63;
c) The remainder when dividing the result by 10: 63 / 10 = 6 remainder 3
d) If the remainder is equal to 0 then use 0 as the check digit, and if not 0 subtract the remainder from 10 to derive the check digit: 10 - 3 = 7.
e) So 036000241457 is a valid UPC because the result in (d) is equal to the Check Digit.

The **`Main`** method will do the followings:

a. It calls the **`VildateUPC`** method to validate the UPC entered by the user;
b. If the the user inputs an invalid UPC, it will repeat step (a) above, until the user inputs a valid UPC.

**Additional Requirement, Assumption and Hints:**

1. **Zero** marks will be given if the program is not compilable;
2. **Zero** marks will be given if you do not following the naming of methods/parameters stated above;
3. You are NOT allowed to use any pre-defined classes except System.Console;
4. You are NOT allowed to use array or any collection classes;
5. You MUST use a loop to scan through the UPC;
6. You may assume the input must contain 12 digits when testing the program.
7. Hints: The following code can convert a charater type digit to an integer:

```
char ch = '9';
int digit = ch - '0'; // digit will store store the integer value 9
```

8. Hints: The following sample code shows you how to get the length of a string and get a character in a position:

```
string str = "This is a string";
int len = str.Length;              // len will be 16
char firstCharacter = str[0];      // firstCharacter will be 'T'
char secondCharacter = str[1];     // secondCharacter will be 'h'
```

**Sample Run #1 (Green text refers to user input):**

```
Enter an UPC >> 123123123123
The UPC is invalid.
Enter an UPC >> 036000241457
The UPC is valid.
```

**Sample Run #2:**

```
Enter an UPC >> 123456789098
The UPC is valid.
```

Selected Answer: Q2.cs

← OK