CSE 174 – Fall 2019
PROGRAM #10: 40 points – Due November 3, by 11:59 p.m.

Outcomes:

- Implement methods for working with arrays.
- Use loops to manipulate array values.

Scoring:

- If you do not submit your source code, your score will be zero.
- If you submit source code that does not compile, your score will be zero.
- If you submit source code without the correct class name (**Program10**) or the correct method names (see below), you will receive at most half-credit for this assignment.
- Deductions will be made for not meeting the usual requirements: Source code that is not formatted according to the usual guidelines, methods that are not commented, and so on

	Full credit	**Partial credit**
Implement getValues() (5 points)	Your method correctly reads values from the input and saves them inside an array and returns the array.	You implemented the method, but with some errors.
Implement the alternatingSum() method (10 points)	Your method returns the correct result.	You implemented the method, but with some errors.
Implement the reverse () method (10 points)	Your method correctly reverses values inside the array and returns a new array.	You implemented the method, but with some errors.
Implement the run() method (10 points)	Your method returns the correct result.	You implemented the method, but with some errors.
Format console output, Comment, file names, short methods, etc. (5 points)	You formatted output as specified, including aligning numbers. Your code has comprehensive comments	Screen output does not match specifications, or not enough comments.

Specifics:

1. Create a class named **Program10**. Your program should have the following 4 methods (other helper methods are allowed, but these four methods must be implemented as specified).
2. You need to define a global scanner object and only use it inside the main method and the getValues method, since other methods do not get any values from the input.
3. **getValues**:
   - This method does not have any input parameter and returns an array of integers.
   - This method prompts the user with a message and gets a positive number. Method shows "Invalid inputs" message for negative numbers and zero, and askes the user to re-enter the number. This positive number is for how many integers needs to be received from the input.
   - This method uses that positive value to get integer values from the input and saves them inside an array.
   - This method returns an array which is filled up with integer values.
   - Finish this method, test it and then start other methods since you will be calling this method from other methods.
4. **alternatingSum:**
   - Read P6.6 at the end of chapter 6.
   - This method does not have any input parameters.
   - This method gets an array of integers by calling geValues() method.
   - This method returns the alternating sum of all elements.
5. **reverse:**
   - Read P6.7 at the end of chapter 6.
   - This method does not have any input parameters.
   - This method gets an array of integers by calling geValues() method.
   - This method returns an array with reversed elements.
   - All reversed elements need to be displayed inside the main method.

6. **Run:**
   - Read P6.12 and P6.13 at the end of chapter 6, and you need to do P6.13.
   - This method does not have any input parameters.
   - This method generates 20 random dice tosses and saves them inside an array.

- This method generates and prints 20 dice values as specified in P6.13 programming exercise (no return value).

Output:

Inside the main method, create a menu and call all methods from the main method.
Optional: you can also create an extra method for the menu and call it inside the main method.
Submit Program10.java file on canvas.

```
>run Program10
PROGRAM#10
1- alternatingSum
2- reverse
3- Run
4- Exit
Enter a number[1-4]: -1
Invalid input!
Enter a number[1-4]: 5
Invalid input!
Enter a number[1-4]: 1

** alternatingSum **
How many integer values: 0
Invalid input!
How many integer values: -2
Invalid input!
How many integer values: 9
Enter the numbers: 1 4 9 16 9 7 4 9 11
The result is: -2

PROGRAM#10
1- alternatingSum
2- reverse
3- Run
4- Exit
Enter a number[1-4]: 2

** reverse **
How many integer values: 9
Enter the numbers: 1 4 9 16 9 7 4 9 11
```

```
The reverse array is: 11 9 4 7 9 16 9 4 1

PROGRAM#10
1- alternatingSum
2- reverse
3- Run
4- Exit
Enter a number[1-4]: 3

** run **
The sequence of numbers is:
1 2 5 5 3 1 2 4 3 (2 2 2 2) 3 6 5 5 6 3 1
```