Overview

In this lab, you will create a $C^{\#}$ console application that generates role-playing game character attributes. The content necessary to complete the Lab is found in Chapter 3 of *Essential* $C^{\#}$ *6.0* by Mark Michaelis, the .NET Framework documentation (in particular the `Random` class), and the *Dungeons & Dragons*® *5th edition System Reference Document*.

As a planning tool and to serve as documentation, you will also submit a UML Activity Diagram that describes the control flow of your application. A tutorial for creating an activity diagram as well as a web-based tool for doing so can be found at LucidChart.

a 20-sided die

Background

In role-playing games (RPGs), the player represents a fictional character. Gameplay occurs as the player makes choices, causing the character to take actions—affecting entities in the game world. How the entities respond is dictated by the game rules, which often incorporate randomness to offset player choices and introduce risk. Random consequent events are not typically uniformly distributed, but shaped by attributes (also known as "stats" or "ability scores") assigned to game world entities, including the player character.

The character creation process used by most RPGs is a variant of the *Dungeons & Dragons*® (D&D) method used since the game was first introduced in 1974. Character stats are randomly determined through dice rolling—in this case, the sum of three rolls of a six-sided die, resulting in a value in the range of 3–18.

The stats used by D&D-derivative games are as follows:

Strength (STR): Strength measures muscle and physical power. Strength also sets the maximum amount of weight a character can carry. A character with a Strength score of 0 is too weak to move.

Dexterity (DEX): Dexterity measures agility, reflexes, and balance. A character with a Dexterity score of 0 is incapable of moving and is effectively immobile (but not unconscious).

Constitution (CON): Constitution represents a character's health and stamina. A character with a Constitution score of 0 is dead.

Intelligence (INT): Intelligence determines how well a character learns and reasons. A character with an Intelligence score of 0 is comatose.

Wisdom (WIS): Wisdom describes a character's willpower, common sense, awareness, and intuition. A character with a Wisdom score of 0 is incapable of rational thought and is unconscious.

Charisma (CHA): Charisma measures a character's personality, personal magnetism, ability to lead, and appearance. A character with a Charisma score of 0 is unable to interact with the world.

Instructions

In this assignment, you will use the *D&D 5th edition* rules for character creation to randomly generate a set of ability scores: **Roll four six-sided dice, drop the lowest value, and sum the remaining three.**

You will also modify these scores according to the specifications of a player race chosen from the *D&D 5th edition* core races. Each of these races affect the ability scores generated in the previous step. You should reference the *Dungeons & Dragons*® *5th edition System Reference Document* to determine how each racial choice affects ability scores. There are two important caveats:

1. You are only responsible for races outlined under the section entitled *Core*, except for Half-Elf and Tiefling. Implement these races for extra credit (see **BONUS** below).

2. Some races have *subraces*. These subraces add additional effects and constraints on ability scores. You are responsible for including subraces listed as "core".

Part I

You will generate a UML Activity Diagram outlining your plan for control flow in this assignment. You may use an online tool such as LucidChart if you wish, but you are not required to do so.

The following specifications must be observed:

- The user will first be presented with a screen that displays six generated scores, labeled as above.
- The user will be offered the opportunity to *reroll* the scores or *keep* them before proceeding to the next step.
- Once the user has kept a set of scores, he/she will be prompted to select a race from a menu of possible choices.
- If (and only if) a chosen race has subraces, the user will be presented with a second menu to select among the appropriate subraces.
- The user will be presented with a final readout, including:
  1. Race OR Subrace
  2. Six labeled ability scores. Each must show the calculation as well as the *final* score, *base* score, and *racial modifier* in the format:

$$LABEL : FinalScore = BaseScore + RacialModifier$$

⋆ **BONUS:** The Tiefling and Half-Elf races have more complex options for attribute customization. Implement them for extra credit.

Part II

Translate your UML Activity Diagram into $C^{\#}$ code. Note that you will need to use looping as well as conditional statements to define the control flow of the application. Although we have not covered methods yet, you may use static methods to organize your code *if you wish*. It will definitely make for more readable code.

Example Output

```
Randomly Generated Scores:
  STR: 13
  DEX: 14
  CON: 11
  INT:  6
  WIS: 11
  CHA: 10

Do you want to (k)eep or (r)eroll? r

====================================
Randomly Generated Scores:
  STR: 15
  DEX: 14
  CON: 12
  INT: 16
  WIS: 14
  CHA:  9
```

```
Do you want to (k)eep or (r)eroll? k

====================================
  1 - Dragonborn
  2 - Dwarf
  3 - Elf
  4 - Gnome
  5 - Half-Elf
  6 - Half-Orc
  7 - Halfling
  8 - Human
  9 - Tiefling

  Select a race: 2

====================================
  1 - Hill Dwarf
  2 - Mountain Dwarf

Select a sub-race: 1

====================================
Hill Dwarf

STR: 15 = 15 +  0
DEX: 14 = 14 +  0
CON: 14 = 12 +  2
INT: 16 = 16 +  0
WIS: 15 = 14 +  1
CHA:  9 =  9 +  0

====================================
```

Submission

You will fork the starter code given in the `repl.it` project, modify it, and share with me. The UML diagram should be submitted to Canvas as well, as an attached file or link.