# COMP 102: Computer & Programming Fundamentals I

## Homework Assignment #2

Program design due on **22 July 1996 in class**

Final program due on **26 July 1996 in class**

**Problem Description:**

Each game in bowling consists of ten frames. In each frame, there is a chance to knock down up to ten pins by throwing a ball in one or two rolls down the bowling alley. To start, you roll the first ball of the first frame. The score for each frame is computed following the rules given below:

- If all ten pins are knocked down, a **strike** is achieved for that frame and a frame score of 10 plus the total number of pins knocked down on the next two rolls is earned.
- If any pins remain standing, you roll a second ball for the frame. After this roll,
  - if all pins are down, you have a **spare**, and a frame score of 10 plus the number of pins knocked down on the next roll.
  - if any pins remain standing after the second roll, then the result is an **open** frame, and the frame score obtained is given by the total number of pins knocked down on the two rolls of the frame.
- If a **strike** or **spare** occurred in the last (the tenth) frame, you make, respectively, two or one additional rolls.

The total score for a game is the sum of the ten individual frame scores.

Write a program to simulate a series of bowling games. Your program should

- ask the user for the number of games in the series.
- allow the user to input score for each roll **or** use a random number generator to generate score for each roll in each game (see description below for **next_score**).
- print out the total score for each game.
- print out the total, minimum, average, and maximum scores for the series of games.

For each game, use a function (**next_score**) to generate a score for the next (and first) roll in the game. That is,

- if the roll is the first roll in a frame, **next_score** returns a value between zero and ten.
- if the roll is the second roll in a frame, **next_score** returns a value between zero and (10 - **r1**), where **r1** is the score obtained in the first roll.

**next_score** should allow two ways for generating the score:

- the score is entered by the user.
- the score is generated by the random number generator **random. random** is a pseudo-random number generator included in **stdlib.h**. It takes one integer argument **n** and returns an integer between 0 and **n**-1 (ref. pp.866 of text).

**Hint: next_score** could be written as a function with three arguments:
**e.g.** int **next_score** ( int **user_or_random**, int **roll1_or_roll2**, int **first_roll_score** )

**Note:** This assignment does not require array and you are **not** allowed to use array in this program.

**Program Design (30%)**

- Specify the input and output of the given problem.
- State any assumptions (if any) that you may have.
- Develop an algorithm for solving the problem using stepwise refinement.

**Program Implementation (60%)**

- Develop a C++ implementation of the above design. Turn in both a hard copy and a disk copy of your program.
  - correctness (40%)
  - program style and documentation (20%)

**Test Cases (10%)**

- List the test cases that you used in testing your program.
- For each test case, justify your choice.

**SAMPLE OUTPUT** (user input is in bold form):

**Example 1**:

Enter number of games:
**2**

Enter " 1" Input scores from the terminal.
"2" Generate random scores.
**1**

Input score for each roll in game 1:
**5 4 6 4 2 4 1 4 3 4 8 1 9 0 10 10 9 0**
Total score for game 1 is 114.

Input score for each roll in game 2:
**9 1 10 7 3 9 0 10 10 5 4 6 4 10 10 10 8**
Total score for game 2 is 199.

Overall results for 2 games:
Total = 313
Minimum = 114
Average = 156.5
Maximum = 199

**Example 2:**

Enter number of games:
**2**

Enter " 1" Input scores from the terminal.
"2" Generate random scores.
**2**

Score generated for each roll in game 1:
0 0 3 0 3 1 5 1 7 3 3 3 1 6 6 0 1 8 7 3 9
Total score for game 1 is 73.

Score generated for each roll in game 2:
10 2 3 10 9 1 8 1 6 3 6 2 7 0 4 0 7 1
Total score for game 2 is 103.

Overall results for 2 games:
Total = 176
Minimum = 73
Average = 88
Maximum = 103

**IMPORTANT NOTE**

Unless otherwise stated, the assignments you submit for grading must represent the results of your own independent effort. Suspected cheating will be referred to the relevant University authorities.

Always clean up the temporary working directory in the hard disk after working on a PC in the computer laboratory. Don't be so generous to let others read and copy your programs. You may get into trouble if another student's submission appears similar to yours.