The exercise: Human behavior prediction

The study of the predictability of human behavior is an intense field of research, mainly for its applications in stock exchanges, in games, in government treatises, sports and also to identify which student will give up the python course Recently, computational methods have been surpassing the human capacity to predict behavior [1].

In this programming exercise, we will implement a program that "predicts" human behavior using a simple game.

The game consists of a human player and the computer. Both choose either the value 0 or the value 1. If both choose the same numbers, the computer earns a point. If the numbers chosen are different, the human player earns a point. This is repeated n times. The winner is the one who accumulates more points at the end of n moves. The strategy to be implemented for the computer to "predict" the human player's bid is based on a simple "machine learning" algorithm.

This learning assumes that every choice the player makes is influenced by his/her previous move. In this way four counts will be made:

> (1) throw00 = count of the number of times the human player chose 0 given that in the previous bid his/her bid was 0,
>
> (2) throw01 = count of the number of times the human player chose 0 given in the bid previous his/her bid was 1.
>
> (3) throw10 = count of the number of times the human player chose 1 given that his/her previous bid was 0.
>
> (4) throw11 = count of the number of times the human player chose 1 given his/her previous bid was 1.

So, the following cases may occur:

If the player's last throw was 0:
1. If throw10 > throw00: then the computer chooses 1
2. If throw10 < throw00: then the computer chooses 0
3. If throw10 = throw00: then the computer chooses randomly 0 or 1.

If the player's last throw was 1:

4. If throw11 > throw01: then the computer chooses 1
5. If throw11 < throw01: then the computer chooses 0
6. If throw11 = throw01: then the computer chooses randomly 0 or 1.

   To avoid cheating, in your code, always make the computer choice 0 or 1 before asking the user for its choice of move 0 or 1.

For the first throw of the computer, there is no previous throw of the player, so computer chooses randomly 0 or 1.

For the random throw, proceed as follows. Generate a random number using the linear congruences method (see Appendix). If the random number is less than or equal to $2^{32}/2$ (=2^{31}), the computer will play 0. Otherwise, it will play 1. To facilitate the structure of your program, generate a new random number at the beginning of each move, and use it only if necessary according to the rules above. If the random number is not used it will be discarded at the beginning of the next move when a new number is generated.

Only when you initialize your python game, it will require an initial seed x0, manually choose an integer value of your choice between 1 and 2^{31}.

After n rounds, wins who gets the most points (player or computer).

Your Individual Programming Exercise (IPE) should receive as input (via keyboard) the number of moves to be made (positive integer) and difficulty of the game (easy or difficult).

In "easy" mode, the computer has no learning, i.e., all computer throws are random, using the method of linear congruences. If the random number generated is less than or equal to $2^{32}/2$ (=2^{31}), the computer will choose 0 and otherwise, 1.

In “difficult” mode, the computer will use the learning method described above.

During the match, your IPE should show graphically how the game is going, i.e. print a bar that illustrates the score of the computer and the score of the player. The score will be simulated by the impression of asterisks.

Example: PLAYER: * COMPUTER: ***

(in this example, the player won once and the machine three times)

At the end of the game (after the n-th0 round), you must print on the screen who was the winner or if there was a tie.

Example 1: You won!

Example 2: Computer won!

Example 3: It was a tie!

Play against the computer and see if you are able to beat it!

See below an example of playing the game using seed 1234. Your program should ask the user if the user wants to play a new game or not.

The output must be exactly the same format as in the example below. The red part will not be in the print massage, they are examples of inputs entered by the player. (Replace "Your Name" with your actual name)

```
Welcome to Human Behavior Prediction by Your Name Please choose the type of game (1: Easy, 2: Difficult):
1 Enter number of moves: 5

ROUND 1 Choose your move for round: 1, (0 or 1): 1 Player Move= 1 Machine Move= 1 . Machine wins! You:
0 computer: 1
PLAYER:

COMPUTER: *

ROUND 2 Choose your move for round: 2, (0 or 1): 0 Player Move= 0 Machine Move= 1 . Player wins! You: 1
computer: 1
PLAYER: *

COMPUTER: *

ROUND 3 Choose your move for round: 3, (0 or 1): 1 Player Move= 1 Machine Move= 1 . Machine wins! You:
1 computer: 2
PLAYER: *

COMPUTER: **

ROUND 4 Choose your move for round: 4, (0 or 1): 0 Player Move= 0 Machine Move= 0 . Machine wins! You:
1 computer: 3
PLAYER: *

COMPUTER: ***

ROUND 5 Choose your move for round: 5, (0 or 1): 0 Player Move= 0 Machine Move= 1 . Player wins!
ou: 2 computer: 3
PLAYER: **

COMPUTER: ***
Machine Wins, Easy Game is Over final result Player: 2 - Computer 3 Do you want to start a new game? Yes
(Y) No (N): N

Total Player Wins: 0 Total Computer Wins: 1
```