CoE 163

Computing Architectures and Algorithms

Machine Exercise 01

Academic Period: 2nd Semester AY 2020-2021
Synopsis: Formulate and profile three different algorithms to compute the number of pins in a game of generalized bowling
Submission Platform: UVLe

Problem Statement

You are a pin boy for a game of generalized bowling in the 1940s. Your job is to manually place bowling pins in place after each frame (three rolls or less, depending on whether the player got a strike or a spare). However, this repetitive task gets boring easily. Luckily, electromechanical systems are all in the rage, and you decided to learn mechanics, mathematics, electric circuits to build an automated pinsetting machine.

A game of the usual ten-pin bowling consists of ten pins arranged in a triangle. The frontmost layer has one pin, two at the next, three at the next, and four at the backmost, for a total of four layers. In generalized bowling, the number of layers is predetermined at the start of the game. Then, the pinsetter determines the number of pins to use such that they can arrange them in a triangle similar to ten-pin bowling. For example, if the number of layers is 3, then we need 6 pins. If it is 6, then we need 21 pins.

You are now faced with the problem of determining the formula for computing the number of pins your pinsetter needs for a given number of layers. Assume that your machine and bowling lane can accommodate an “infinite” number of pins (i.e. millions of pins). Since it will be your first electromechanical machine, you are zealous to formulate three algorithms and choose the fastest one. To help with your decision, the bowling alley where you work is open for 8 hours every day, and a game of 12-frame bowling lasts 15 minutes on average.

Input

Input to your electromechanical machine would be the number of layers needed for the game N.

Output

Your program should be able to compute the number of pins needed to set-up a bowling game with Nlayers.

Example

Input: 3 *Output:* 6
Input: 4 *Output:* 10

Additional Description/Requirements

In addition to the algorithm, we expect you to formulate three different ways to solve the problem. Time each of your algorithms by first formulating a function for the pinsetter algorithm. Then, enclose that function call in a loop that runs at least 384 times, corresponding to the number of times a bowling lane is used every day. The input to the function can be the index of the loop (e.g. if the loop is now on its 50th iteration, then the number of layers entered into the algorithm is 50). For example, in an ancient language named Turbo Pascal, this is how you may write your timed algorithm.

```
program Pinsetter;

var i: integer;

begin
  { TIMER start }
  for i:= 1 to 1440 do begin
     { PINSETTER ALGORITHM here }
  end;
  { TIMER end }
end.
```

Assume that in the 1940s, the programming languages of today exist because you're that good in inventing such. Hence, use C/C++, Python, or any programming language you are comfortable using. Profile each of your three algorithms using your preferred profiler.

Being a responsible pinsetter, you also would like to write a short journal entry on which algorithm you chose to load into your machine. Explain why that algorithm was correct, how long did it take to run at least 384 times, and why it ran the fastest among all of the three.

Upload both your algorithm (as a single source code file; in TXT if the system does not support the file extension) and your short journal (PDF or TXT file) to a futuristic information repository named UVLe, which your humble pinsetter machine can apparently communicate to!

Grading Rubric

45% Algorithm code and correctness, for all three algorithms
30% Profiling code for all three algorithms
25% Short journal