



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Assignment 1: Report

COMPUTATIONAL THINKING

Shaun Ross Holborow

1704229

Course: Computer Networks and Cyber Security

Faculty name: Dr Tim Bashford

Date of Submission: 27/04/2018



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Contents

1. Essay	4
2. References	7
3. Appendices	8
3.1. Appendix A – Exact Code	8
3.2. Appendix B – Screenshots of Coding	10

Declaration page:

The work of this essay has been completed under the supervision of Dr Tim Bashford. I declare that this report is the product of my own work and is not being submitted for any other degree etc.

1. Essay

This essay will investigate numerous programming concepts and consists of looking at different approaches to the languages available on the current market. The programming software in question for investigating is Python and will also be used for the subsequent tasks of developing a program to produce a lottery ticket process. This essay will consider what compiled and interpreted programming languages are available and their differences (pros and cons) as well as the "goto" statement; this will cover potential dangers/difficulties of utilising it and how it is used with control flow statements. Further to this, random and pseudorandom numbers generated by computers and how these random numbers are generated and what the meaning of a pseudorandom number is will be discussed. Finally, this essay will finish with a discussion of what a lottery number generator is and how it works coding wise when presented to the user.

Programming languages are mainly comprised of 2 types of implementation and are used entirely on their capabilities and function for certain tasks. A compiled programming language is a type of implementation that uses what are known as compilers which are made to translate or generate machine code from source code and example of this would be C. The positives of using this method of translation are that once its compiled can be immediately utilised by any user on demand, It is typically optimised for CPU's meaning a faster program processing speed and provides the developer with a "safe guard" as it protects their intellectual property that they were responsible for developing from being tampered with by keeping the source code private.

However, while the negatives are that they are not typically cross-platform friendly meaning it may only work on certain systems, it is non flexible in the sense that the code needs to also be recompiled for other CPU's besides the one it was originally created on and in order to effectively test said program and extra step must be take each time for it to run correctly. An interpreted programming language is a type of implementation that executes steps directly and openly without prior compiling of a program into machine language.

The positives of using this method of translation are that it's much more cross-platform friendly in that the machine code is not send but only the source is which is left to the computer to process, is simpler to test because all that is required is to implement selected source code which is then converted by the interpreter to run the program and is simpler to debug any issues as the source code is not locked from the user.

However, the negative side of this is that in order to run or edit said program an interpreter is required in order gain access to the code or run it, is typically slower than compiling because the source code has to interpreted each time the program is activated and the source code is not private meaning anyone can utilise it from their own reasons without being restricted.

Most, if not all programming languages use statements as part of their coding. However, many still do not use certain statements particularly the "goto" statement which is not used for various reasons though they typically word it as being too dangerous or difficult to work with.

The apparent reasoning behind this is that it can potentially create bad control flow for other programmers trying to edit or fix it thus making them less appreciative of the code implemented. Besides these reasons, the "goto" statement has been essentially superseded by various control flow statements such as "continue" etc.

One of the most common uses for machines are machines programmed to deliver or provide results dependant entirely on a specific number, result varies depending on the number listed or generated but these typically known as "random number generators" which are typically in places like casinos for gambling machines where money is inserted to provide a random result in the hopes of receiving the winning digits to cash in on the prize.

Most random number generators work based on computing whereas before digital computing was present this was done in its simplest forms using things like dice, coin flipping or roulette wheels which are still used today though not in the traditional sense but more for entertainment than in serious processing of important matters such as money etc. With digital computation being the norm, computational methods are mostly used instead to work out things using various programs which provide more speed and efficiently for processing outcomes or number results. These generators work via mathematical algorithms which utilise a specific fixed seed and in the case of pseudo random number generators, "linear congruential generator" is used which works based on recurrence to generate the numbers. These random number generators consist of two types, one of which is known as a "Pseudo Random Number" generator which is a type generator that does not show true random numbers but rather something close or similar to what a true result would be and is commonly utilised.

An example of this being used is with interpreted high-level programming languages like Python which allow users to create various programs in this case a simple lottery number generator as shown in the following screenshots below. The code works by utilising a list of a range that goes from 1 to 49 for each individual ball including the bonus ball a random number is generated from said list via random shuffle function (this shuffles the sequence in place) and this is repeated for each ball. These results are then presented as print or "text" which first states that the lottery selection is being started, then each ball is stated with the following number assigned from selection and finally a message stating the selection process is complete.

This form of coding/programming uses what is commonly known as "loops" which allow the user to select a portion of code that they wish to repeat a certain number of intervals and this is done in an orderly fashion.

This essay was written in order to investigate computational thinking involved with the types of programming languages, coding statements, random number generation and the meaning behind the common formation of random numbers being pseudo random number generation. In this essay, the primary body talks about the differences between compiled and interpreted programming languages showing that they both have reason for being utilised despite their positives and negative attributes.

The learning experience achieved from this topic of research was that many types of programming languages utilise different but similar functions related to things such as coding

statements and random number generation (most common type utilised being pseudorandom generation instead of while also learning that compiled and interpreted languages are not better than one another but rather are good at specific things that the other type what not be able to do effectively if at all.

Then the validity of the "goto" statement is discussed which despite being almost entirely replaced nowadays by control flow statements due to their flexibility and less hazardous nature, "goto" statement is still utilised for more specific tasks where it shines in efficiency. The essay then moves forward to discuss the topic of random number generators on how these random number are created and specifically what the meaning behind "pseudorandom number" is.

Looking at this aspect of the research specifically, what is achieved from this topic of research was that prior to proper control flow statements, most programming languages utilised the "goto" statement for one-way transfers of control to other lines of code, but this was left behind because of how hard it would make the tracking process of control flows within the coding of programs.

Also, most if not all random number generators work via pre-calculated/determined tables to present numbers as being "random" thus are named "pseudorandom number generators" and are not considered real random number generators due to its inherit nature.

Finally, the code and workings of the lottery number generator are discussed with topics such as what they are, how they work and what they are used for etc. The learning experience achieved from this topic of research was that there are essentially two types of random number generators of which are "pseudorandom number generators" and "true random number generators" where pseudorandom number generators work via pre-determined tables of numbers where as true random number generators work via extracts numbers from physical sources such as radioactive sources which are unpredictable thus creating unpredictable results of numbers.

PRNG (pseudorandom number generators) are used seemingly for activities like casino gambling machines where as TRNG (true random number generators) are used for physical phenomena such as radioactive sources as mentioned previously.

2. References

[1]"How Computers Generate Random Numbers", *Howtogeek.com*, 2018. [Online]. Available: <https://www.howtogeek.com/183051/htg-explains-how-computers-generate-random-numbers/>. [Accessed: 22- Apr- 2018].

[2]M. Haahr, "RANDOM.ORG - Introduction to Randomness and Random Numbers", *Random.org*, 2018. [Online]. Available: <https://www.random.org/randomness/>. [Accessed: 24- Apr- 2018].

3. Appendices

3.1. Appendix A – Exact Code

```
# Lottery machine
```

```
import random
```

```
lottery = list(range(1,49))
```

```
numbers = []
```

```
for i in range (1):
```

```
    random.shuffle(lottery)
    a = lottery.pop()
    numbers.append(a)
```

```
for i in range (1):
```

```
    random.shuffle(lottery)
    b = lottery.pop()
    numbers.append(b)
```

```
for i in range (1):
```

```
    random.shuffle(lottery)
    c = lottery.pop()
    numbers.append(c)
```

```
for i in range (1):
```

```
    random.shuffle(lottery)
    d = lottery.pop()
    numbers.append(d)
```

```
for i in range (1):
```

```
    random.shuffle(lottery)
    e = lottery.pop()
    numbers.append(e)
```

```
for i in range (1):
```

```
    random.shuffle(lottery)
    f = lottery.pop()
    numbers.append(f)
```

```
for i in range (1):
```

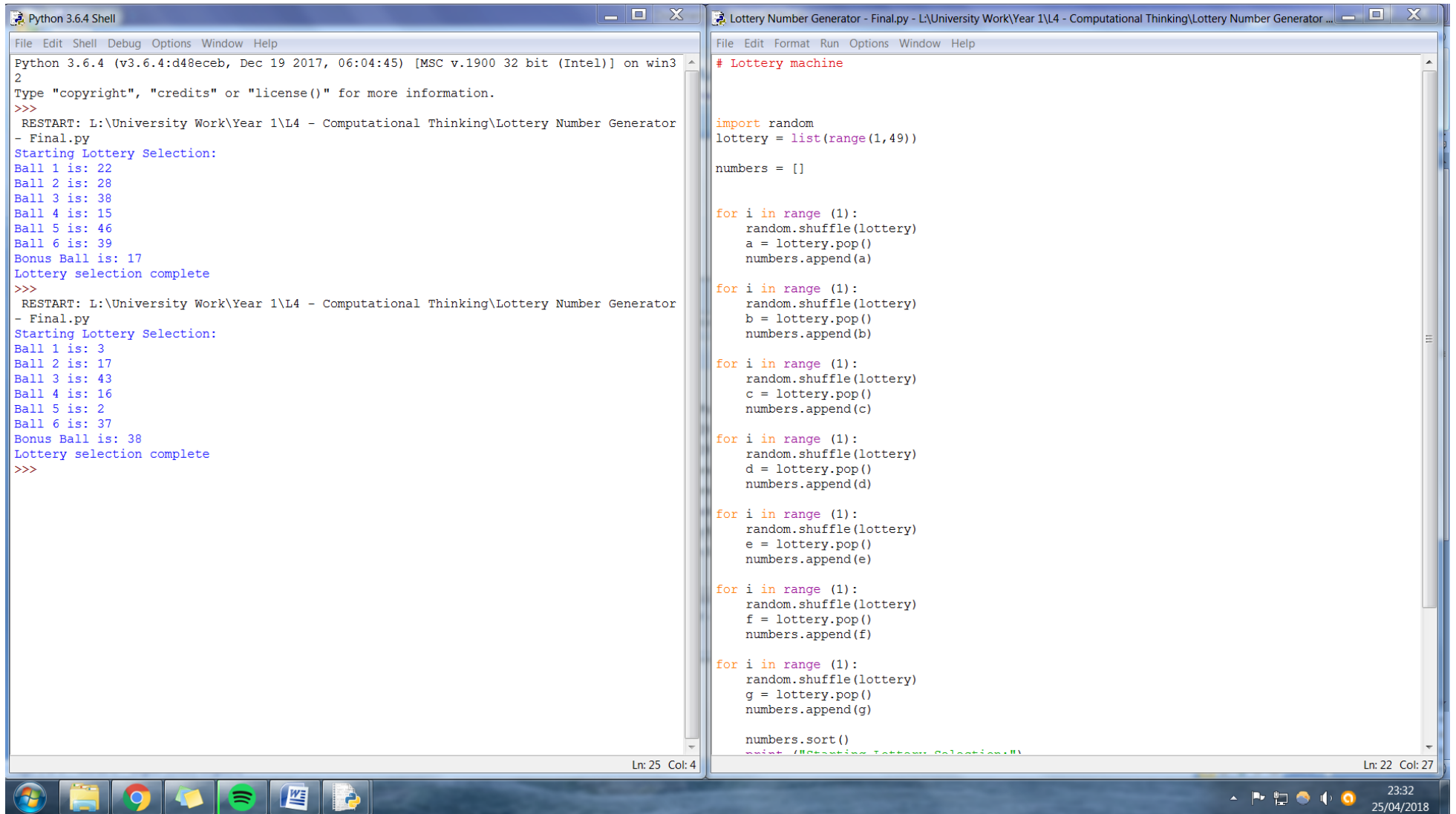
```
    random.shuffle(lottery)
    g = lottery.pop()
    numbers.append(g)
```

```
numbers.sort()
```

```
print ("Starting Lottery Selection:")
```

```
print ("Ball 1 is:", a)
print ("Ball 2 is:", b)
print ("Ball 3 is:", c)
print ("Ball 4 is:", d)
print ("Ball 5 is:", e)
print ("Ball 6 is:", f)
print ("Bonus Ball is:", g)
print ("Lottery selection complete")
```

3.2. Appendix B – Screenshots of Coding



```
Python 3.6.4 Shell
File Edit Shell Debug Options Window Help
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: L:\University Work\Year 1\L4 - Computational Thinking\Lottery Number Generator
- Final.py
Starting Lottery Selection:
Ball 1 is: 22
Ball 2 is: 28
Ball 3 is: 38
Ball 4 is: 15
Ball 5 is: 46
Ball 6 is: 39
Bonus Ball is: 17
Lottery selection complete
>>>
RESTART: L:\University Work\Year 1\L4 - Computational Thinking\Lottery Number Generator
- Final.py
Starting Lottery Selection:
Ball 1 is: 3
Ball 2 is: 17
Ball 3 is: 43
Ball 4 is: 16
Ball 5 is: 2
Ball 6 is: 37
Bonus Ball is: 38
Lottery selection complete
>>>
```

```
Lottery Number Generator - Final.py - L:\University Work\Year 1\L4 - Computational Thinking\Lottery Number Generator ...
File Edit Format Run Options Window Help
# Lottery machine

import random
lottery = list(range(1,49))

numbers = []

for i in range(1):
    random.shuffle(lottery)
    a = lottery.pop()
    numbers.append(a)

for i in range(1):
    random.shuffle(lottery)
    b = lottery.pop()
    numbers.append(b)

for i in range(1):
    random.shuffle(lottery)
    c = lottery.pop()
    numbers.append(c)

for i in range(1):
    random.shuffle(lottery)
    d = lottery.pop()
    numbers.append(d)

for i in range(1):
    random.shuffle(lottery)
    e = lottery.pop()
    numbers.append(e)

for i in range(1):
    random.shuffle(lottery)
    f = lottery.pop()
    numbers.append(f)

for i in range(1):
    random.shuffle(lottery)
    g = lottery.pop()
    numbers.append(g)

numbers.sort()
print("Starting Lottery Selection:")
```