A red sign with white text

Description automatically generated

BSc Computer Science

**Module Title**

Advanced Algorithms

**Assessment Title**

Mini Report

**Assessment Weighting**

5% of the module mark

**Student Name**

Reece Turner

(09/05/2024)

Contents

[1. Tasks 3](#_Toc166147496)

[1.1 Password Generator 3](#_Toc166147497)

[1.2 Longest Substring 3](#_Toc166147498)

[1.3 Parallelism Programming 3](#_Toc166147499)

[1.4 Cheapest Train Tickets 3](#_Toc166147500)

[2. Appendix 4](#_Toc166147501)

[Figure 1: Password Generator 4](#_Toc166147502)

[Figure 2: Longest Substring 5](#_Toc166147503)

[Figure 3: Parallelism Programming 5](#_Toc166147504)

[Figure 4: Cheapest Train Tickets 5](#_Toc166147505)

# 1. Tasks

## 1.1 Password Generator

For this task I had to write an algorithm that generates passwords within a rule set. In **Figure 1** my rule set can be seen within the flowchart’s algorithm decision statements. To perform such decisions for each password I implemented Tail Recursive Optimisation (TCO). This algorithm effectively calls its entry point until a task is complete. In contrast to typical recursion, TCO will handle memory leaks by ending the recursion if criterion is met to prevent overflow. However, given the size of the input, , and the number of categories, , the time complexity of this algorithm would be therefore expressing exponentiality showing that smaller inputs are an optimal use case.

A computer screen with white text

Description automatically generatedA screenshot of a graph

Description automatically generated

*Figure 1.1: Execution of the program with password length of 4.*

In **Figure 1.1** the algorithm demonstrated complexity by averaging the time over 5 runs on a password length of 4. You can see through the numbers its permuting through each result in sequence to find the product. As a result, it can be slower, especially ran in a Python environment; however, this method was more effective over implementing the “itertools” library ‘product’ due to its built-in functionality not being able to consider rule sets.

## 1.2 Longest Substring

## 1.3 Parallelism Programming

## 1.4 Cheapest Train Tickets

# 2. Appendix

## Figure 1: Password Generator

A diagram of a diagram

Description automatically generated

## Figure 2: Longest Substring

## Figure 3: Parallelism Programming

## Figure 4: Cheapest Train Tickets