


COSC 3360 - Fundamentals of Operating Systems

[Dashboard](#) / [My courses](#) / [COSC3360SP2022](#) / [PROGRAMMING ASSIGNMENTS](#) / [Programming Assignment 3](#)

 Description

 [Submission view](#)

Programming Assignment 3

 **Due date:** Monday, 2 May 2022, 11:59 PM

 **Requested files:** main.cpp ( [Download](#))

Type of work:  Individual work

Similarity Threshold: 90%

Objective:

This assignment will introduce you to interprocess synchronization mechanisms in UNIX using named POSIX semaphores, pthread mutex semaphores, and pthread condition variables.

Problem:

For this assignment, you will modify your solution for [programming assignment 1](#) to comply with the restrictions explained below.

Given an alphabet (where each symbol is represented by a character and a decimal value with the code of the symbol), you need to implement a fixed-length code decompressor based on the following steps:

- Read the contents of the input file using input redirection (STDIN). The format of the input file is as follows:
 - An integer value representing the number of symbols in the alphabet
 - n lines (where n is the number of symbols in the alphabet) with a char representing the value of the symbol and an integer value representing the symbol's code (in decimal notation).
 - A string representing the compressed message (sequence of bits).

Given the previous format, the following file represents a valid input file:

```
3
a 2
b 4
c 5
010010010100101101101
```

- Calculate the number of bits of the fixed-length codes based on the value of the greatest base 10 code in the alphabet. For decimal values greater than zero, you can use the following formula to calculate the number of bits of the fixed-length codes:
 $\text{ceiling}(\log_2(\text{greatest_base_10_code_in_the_alphabet} + 1))$
- Create n child threads (where n is the number of symbols in the alphabet). Each child thread executes the following tasks:
 - Determines the binary representation of the symbol's code.
 - Determines the frequency of the symbol in the compressed message.

- Print the character, binary code, and frequency of the assigned symbol.

- Create m child processes or threads (where m is the number of characters in the decompressed message) to determine each character of the decompressed message.
- Print the decompressed message.

Given the expected input, the expected output is:

```
Alphabet:
Character: a, Code: 010, Frequency: 3
Character: b, Code: 100, Frequency: 1
Character: c, Code: 101, Frequency: 3
Decompressed message: aaabccc
```

NOTES:

- You can safely assume that the input files will always be in the proper format.
- You cannot use global variables. A 100% penalty will be applied to submissions using global variables.
- You must define the critical sections following the guidelines that we discussed in class.
- You must use POSIX threads. A penalty of 100% will be applied to submissions using a thread library other than the pthread library.
- You can only use named POSIX semaphores, pthreads mutex semaphores, or pthreads condition variables to achieve synchronization. Using pthread_join or sleep to synchronize your threads is not allowed. A penalty of 100% will be applied to submissions using the previous system calls to synchronize the child threads.
- You cannot use different memory addresses to pass the information from the parent thread to the child threads.
- You must use the output statement format based on the example above.

Requested files

main.cpp

```
1 // Write your program here
```

[VPL](#)

[◀ Programming Assignment 2](#)

Jump to...