COSC 3360 - 24967 - Fundamentals of Operating Systems

Available from: Thursday, 16 February 2023, 2:30 PM
Due date: Thursday, 16 February 2023, 3:50 PM
Requested files: main.cpp (Download)
Type of work: Individual work

The exam will close at 3:50 PM. You must save your work before 3:50 PM.

The multithreaded string to decimal transformation.

You must complete the program below (using POSIX threads) that creates n threads to transform a number from string to decimal (where n is the number of digits of the string to transform). Each child thread will store the value of the operation digit * 10 ^ position (where position represents the location of the digit in the string). The position value is represented as follows:

0 = Ones, 1 = Tens, 2 = Hundreds, 3 = Thousands, ...

Given the string "1234", the multithreaded function creates 4 child threads where:

Child Thread 1 calculates 1 * 10 ^ 3 = 1000; Child Thread 2 calculates 2 * 10 ^ 2 = 200; Child Thread 3 calculates 3 * 10 ^ 1 = 30; and Child Thread 4 calculates 4 * 10 ^ 0 = 4.

Finally, the main thread accumulates the results from the child threads to represent the string as an integer value:

1000 + 200 + 30 + 4 = 1234

The input of the program will be a string representing the number:
1234

Given the previous input, four child threads will be created, and the expected output by the parent thread is:

The string "1234" is equal to 1234

Notes:

- Not using POSIX threads will translate into a penalty of 100%.
- Follow the instructions provided in the template file to complete your solution.

- You can always assume that the input will be valid.

Requested files

main.cpp

```cpp
#include <pthread.h>
#include <iostream>
#include <unistd.h>
#include <string>
#include <cmath>

struct term
{
    char digit;
    int pos;
    int result;
};

void *strtodec(void *void_ptr)
{
    term *term_ptr = // cast the void pointer to a struct term pointer
    term_ptr->result = // calculate the value for the term (digit_converted_to_int * 10 ^ position). You can use the pow function.
    return nullptr;
}

int main()
{

    std::string number;
    std::cin >> number;
    int size = number.length();
    pthread_t *tid = new pthread_t[size];
    term * arg= new term[size];

    for (int i = 0; i < size; i++)
    {
        arg[i].digit = // assign the value of the digit from the input string as a character (from left to right).
        arg[i].pos = // assign the value of the position for the digit (0 = Ones, 1 = Tens, 2 = Hundreds, 3 = Thousands, ...).
        if (/*call pthread create here*/)
        {
                fprintf(stderr, "Error creating thread\n");
                return 1;
        }
    }
    //call pthread_join here
    int decnumber = 0;
    for (int i = 0; i < size; i++)
        decnumber = decnumber + arg[i].result;
    std::cout << "The string \"" << number << "\" is equal to " << decnumber << std::endl;
    delete [] tid;
    delete [] arg;
    return 0;
}


```

VPL