CS351 Lab Extra Credit

Midterm Programming Questions

Instructions:

- *Assigned date: Tuesday April 5th, 2022*
- *Due date: 11:59PM on Friday April 8th, 2022*
- *Maximum Extra Credit Points towards midterm exam score: 10*
- ***This lab must be done individually***
- ***Please post your questions to BB***
- ***Only a softcopy submission is required; it will automatically be collected through GIT at the deadline; email confirmation will be sent to your HAWK email address; late submission will be penalized at 10% per day; your submission will not be graded until you have submitted a single page PDF file with your name, email, A#, assignment number, and URL to the gitlab repo***

1. **(2 points) Hash:** Implement a string hash computation that computes a hash on the 1st command line argument (a single string without spaces). You must implement 2 functions, including main() and hash(). The hash function should compute the hash as follows: take the ASCII decimal value of each character, and add all of the values together to get an unsigned long hash value. This hash value should be printed to the screen. The function declarations are included, and they must be followed exactly to receive full points.

   Usage:
   ./hash <string>

   For example, if you type:
   ./hash awesome
   753

   ./hash cs351
   367

2. **(3 points) bsort:** Implement a bubble sort program that will read from a file "10numbers.txt" from the current directory a list of intergers (10 numbers to be exact, e.g. 1 3 2 8 4 9 8 3 10 14), and then sort them, and print them to the screen. You should read the numbers from the command line, and use redirection to read data from the file through standard input. You can assume the input data will only have 10 numbers in the file (no more, or less), and that the numbers will be valid integers. You must implement 3 functions, including main(), bubblesort(), and swap(). The function declarations are included, and they must be followed exactly to receive full points.

   Usage:
   ./bsort

   For example, if you type:
   ./bsort < 10numbers.txt
   Enter 10 integers
   Sorted in ascending order: 1 2 3 3 4 8 8 9 10 14

3. **(5 points) Parallel:** Implement the parallel utility that has 3 command line arguments:
   ./para <processes> <cmd> <args>

   For example, if you type:

```
./para 4 sleep 10
process id = 23361
process id = 23362
process id = 23363
process id = 23364
Child 23361 terminated with status: 0
Child 23362 terminated with status: 0
Child 23363 terminated with status: 0
Child 23364 terminated with status: 0

./para 2 ls
process id = 23421
process id = 23422
Makefile        bubblesort.c    hash.c      parallel.c
bsort           hash            para        pp2.txt
Makefile        bubblesort.c    hash.c      parallel.c
bsort           hash            para        pp2.txt
Child 23421 terminated with status: 0
Child 23422 terminated with status: 0
```

   Implement the parallel utility that has 3 command line arguments. For example “para 4 sleep 10” will execute 4 copies of the sleep process for 10 seconds. Your program should allow an arbitrary number of command line arguments for the process to be run in parallel. The parent process should wait for all child processes to finish before exiting back to the shell. If the format of the command is not recognized (e.g. the first argument is not an integer, or there are not enough arguments given to determine the command to run), an error should be displayed.