COSC 350 System Software Midterm #1-2

10/09/2020

Name: Jacob Duncan

1. (5 pt) What will be the permission for files **foo** and **bar** with following program?

#include <unistd.h>

#include <fcntl.h>

#include <ctype.h>

int main ()

{

umask(0200);

if (creat("foo",S\_IRUSR |S\_IWUSR|S\_IRGRP|S\_IWGRP|S\_IROTH|S\_IWOTH) <0)

return 1;

umask(0440);

if (creat ("bar",S\_IRUSR |S\_IWUSR|S\_IRGRP|S\_IWGRP|S\_IROTH|S\_IWOTH) <0)

return 1;

return 0;

}

**Foo will have read and write permission for group and other, but only read permission for user. -r—rw-rw-**

**Bar will have read and write permission for other, but only write permission for user and group. –w—w-rw-**

1. (15 pt.) Write a C main function that takes one command-line argument, the name of an input file. The input file contains exactly one integer spread out over a single line of up to 80 characters. For example, the integer 3579 is embedded in the line az3mqrm5t?7!z\*&gqmzt9v. Your program uses system calls to do the following:
   1. open and read each character from the input file, accumulating the discovered digit characters into a character array (c-string). (**Do not use** isdigit (int c) function to check number charcter)
   2. Convert the string to an integer (**Do not use** atoi function).
   3. Add 10 to the integer
   4. convert the sum back to a string (use function convIntToStr() function )
   5. Use a **system call** to write the string to standard output.

#include <unistd.h>

#include <fcntl.h>

#include <ctype.h>

#include <stdio.h>

#include <string.h> // you need for strlen() method

int convIntToStr(char\* str, int x);

int convStrToInt(char\* str, int x);

int main(int argc, char\* argv[]) {

if(argc != 2) {

printf(“ERROR: must pass one argument!\n”);

return 0;

}

int inputFile;

if((inputFile = open(argv[1], O\_RDONLY)) < 0) {

printf(“Open %s file error\n”, argv[1]);

return 1;

}

int size = lseek(inputFile, 0, SEEK\_END);

lseek(inputFile, 0, SEEK\_SET);

char buf, str[size];

int curr = 0;

while((read(inputFile, &buf, 1)) == 1) {

if(buf >= 48 && buf <= 57) {

str[curr] = buf;

curr++;

}

}

close (inputFile);

int num = convStrToInt(str, curr);

num+=10;

convIntToStr(str, num);

write(1, str, curr);

return 0;

}

int convIntToStr(char\* str, int x)

{

sprintf(str, "%d", x);

return(strlen(str));

}

int convStrToInt(char\* str, int x) {

int num = 0;

for(int i = 0; i < x; i++) {

num = (num \* 10) + (str[i] – 48);

}

return num;

}

1. (15 pt.) Write a C program named “**evenonly.c**” that reads sequence of integers on the command line and prints their even value sum on the screen. Exit the program with an appropriate error message under the following error condition: if there is no integer input on the command line. Don’t use itoa() function

Ex) for ./evenonly 12 33 44 9 14

output: The sum of even arguments is 70

You need define function **int st\_to\_int(char \*char)** which will convert a c-sting to an integer.

#include <stdio.h>

#include <stdlib.h>

int st\_to\_int(char \*c);

int main(int argc, char \*argv[]) {

if(argc <= 1) {

printf(“You need to enter numbers!”);

return 0;

}

int in[argc];

for(int i = 1; i < argc; i++) {

in[i] = st\_to\_int(argv[i]);

}

int sum = 0;

for(int j = 1; j < argc; j++) {

int curr = in[j];

if(curr % 2 == 0) {

sum += curr;

}

}

printf(“The sum of even arguments is %d\n”, sum);

}

int st\_to\_int(char \*c) {

int index = 0;

int num = 0;

while(c[index] != ‘\0’) {

num = (num \* 10) + (c[index] – 48);

index++;

}

return num;

}

1. (10 pt.) Redo problem 9 with Shell Script. Use for loop. Do not use double parenthesis (()) .

#!/bin/bash

if [ $# -lt 1 ]; then

echo “Pass at least 1 numerical argument”

exit 1

else

sum=0

for i in $\*

do

if [ `expr $i % 2` -eq 0 ]; then

let sum=sum+$i

fi

done

fi

echo “The sum of even argument is $sum”

exit 0

1. (5 pt.)There are four components in C System environment: Text editor, preprocessor, compiler and Linker. Briefly describe roles of each component.

* Text editor – Where regular files can be written and edited and where users write code.
* Preprocessor – processes include-files, conditional compilation, etc. The first pass on a C compilation, also searches for header files in the standard location.
* Compiler – The second pass. Takes the results from the preprocessor, and the source code, and turns it into code the computer can understand (object code).
* Linker – The final stage of compilation. Takes libraries and object files and combines them to produce an executable which the user can run to see the output of their program.