COSC 350 System Software Midterm #1-1

10/07/2020

Name: Jacob Duncan

1. (5 pt.)Explain what following program does?

#include <stdio.h>

#include <unistd.h>

#include <fcntl.h>

int main()

{

int fd = open("my.txt", O\_WRONLY|O\_CREAT, S\_IRUSR|S\_IWUSR);

if (fd < 0)

return 1;

printf("How are you?\n");

if(dup2(fd,1) < 0)

return 1;

printf("I am fine Thank you. How about yourself?\n");

close(fd);

return 0;

}

**The program opens a file named “my.txt” and if the file descriptor is less than 0, return 1. If file descriptor is not less than 0, ask the user “how are you?”, if we reassign the file descriptor of fd to 1 using dup2, and the value returned is less than 0, return 1, else we print “I am fine Thank you. How about yourself?” and we close the file with fd filedescriptor. End the program with return 0.**

1. (5 pt.) What will be displayed for each of the following sequences of shell commands?

W=K

A=W

B=\$$A

echo $B

**$W**

W=K

A=W

eval B=\$$A

echo $B

**K**

1. (15 pt.) Write C code which pass input (text file) and output file name as command line arguments. Open the input file as read only and open output file with mode rw-rw-rw. Your program encodes each character to ASCII code number and writes to output file. You need consider a space and end of line. You need consider argument number error and open file error. You must not use any library function to convert a character to ASCII number.

**(tip1: use dup2 and type coercion) (tip2: end of line has ASCII number is 10).**

ex)

input file output file

65 65 32 66 66

67 67 32 68 68

AA BB

CC DD

**#include <unistd.h>**

**#include <fcntl.h>**

**#include <stdlib.h>**

**#include <stdio.h>**

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

**if (argc != 3) {**

**printf(“ERROR: Argument number not valid. Expecting input and outputfile”);**

**return 1;**

**}**

**int inputFile, outputFile, num;**

**char buf;**

**inputFile = open(argv[1], O\_RDONLY);**

**if(inputFile <0){**

**printf(“ERROR: Input file error!”);**

**return 1;**

**}**

**umask(0);**

**outputFile = open(argv[2], O\_RDWR | O\_CREAT | O\_EXCL, 0666);**

**if(outputFile <0){**

**printf(“ERROR: Output file error!”);**

**return 1;**

**}**

**if(inputFile > 0 && outputFile > 0) {**

**dup2(outputFile, 1);**

**while((num = read(inputFile, &buf, 1)) > 0) {**

**if((int)buf == 10){**

**printf(“\n”);**

**else if(buf == ‘ ‘) {**

**printf(“%d “, (int)buf);**

**}**

**else if (buf != “ “) {**

**num = (int)buf;**

**printf(“%d “, curr);**

**}**

**}**

**} else {**

**printf(“ERROR: Error reading input and/or output file!”);**

**}**

**close(inputFile);**

**close(outputFile);**

**return 0;**

**}**

1. (10 pt.) Write a C program that takes two command-line arguments: input file name and output file names. This program read input from the input file and writes in the output file in reverse order without any numerical characters (numerical character must be skipped).

Created output file mode will be rw-rw-rw-. You need consider argument number error and an input file error.

#include <unistd.h>

#include <fcntl.h>

#include <stdlib.h>

#include <stdio.h>

#include <sys/stat.h>

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

**if (argc != 3) {**

**printf(“ERROR: Argument number not valid. Expecting input and outputfile”);**

**return 1;**

**}**

**int inputFile, outputFile, curr;**

**char buf;**

**inputFile = open(argv[1], O\_RDONLY);**

**if(inputFile <0){**

**printf(“ERROR: Input file error!”);**

**return 1;**

**}**

**umask(0);**

**outputFile = open(argv[2], O\_RDWR | O\_CREAT | O\_EXCL, 0666);**

**if(outputFile <0){**

**printf(“ERROR: Output file error!”);**

**return 1;**

**}**

**if(inputFile > 0 && outputFile > 0) {**

**int size = lseek(inputFile, 1, SEEK\_END);**

**int curr = -1;**

**while(curr >= -(size)) {**

**lseek(inputFile, curr, SEEK\_END);**

**read(inputFile, &buf, 1);**

**if(buf >= ‘0’ && buf <= ‘9’) {**

**printf(“”);**

**} else {**

**write(outputFile, &buf, 1);**

**}**

**curr--;**

**}**

**} else {**

**printf(“ERROR: cannot read input and/or output file!”);**

**}**

**close(inputFile);**

**close(outputFile);**

**return 0;**

**}**

1. (10 pt.) Write a bash script which checks each of file’s type in the current directory and
   1. If a file is c or c++ program file, compile
   2. If a file is text file ( .txt), display content of text file on the screen.
   3. Other files: just display “ file is not c or c++ or text file” on the screen

Use for loop and case statement (Do not use (()).

#!/bin/bash

for file in \*; do

case $file in

\*.c) exec gcc $file;;

\*.cpp) exec g++ $file;;

\*.txt) exec cat $file;;

\*) echo “$file is not c or c++ or text file”;;

esac

done

exit 0

1. (5 pt.) The following is the output of command “ls –l”

drwxr-xr-x 2 separk users 512 Sep 16 12:47 csc350/

-rw------- 1 separk users 43008 Jan 6 2003 snap.doc

Answer the following questions:

1. Explain the meaning of “d” and “–” at the beginning of the two lines.

**d means that it is a directory**

1. What are the permissions for group for csc350?

**Read and executable permissions for group for csc350**

1. Give a command to add executable permission to others for snap.doc.

**chmod o+x snap.doc**

1. Give a command to remove read permission for group for csc350.

**chmod g-r csc350**

1. Explain the difference between the following two commands.

% ls –l | wc

% ls –l > wc

**The first command pipes the ls -l command into wc command giving you back the word count output to the terminal.**

**The second command redirects the output of ls -l to a file name wc.**