**CS2750 System programming and Tools Fall 2018**

**Some Answers to** **TEST #2 [200 points]**

-------------------------------------------------------------------------------------------------------------

*This is an open-books/notes, closed-computers test. If you are using an electronic textbook, an instructor may check during the test what is on your screen. The test should be completed within 75 minutes.*

1. Suppose that you have to work with the file Data. Write a single sed command to do each of the following:
2. [5 points] Print all lines from the file Data containing an upper-case character **S** followed by either **h** or **u** (both lower-case).

sed -n ‘/S[hu]/p’ Data

1. [10 points] In all lines from the file Data starting from #10 to the line containing a pattern Linux, replace the end of the line with the sequence END. (a dot must also appear).

sed -n ’10,/Linux/s/$/END\./p’ Data

1. Now you work with the file NewData, where each field has the following layout:

John;Miller;10/24/1997;38267;1278

Write a single awk command to do each of the following:

1. [6 points] For each line containing a pattern Jo, print the number of this line and its 3d field.

awk -F; ‘/Jo/{print $NR, $3}’ NewData

1. [9 points] For each line containing a pattern /2 print the second field and the sum of the values in the last two fields.

awk -F; ‘/\/2/{print $2, $4+$5}’ NewData

1. (a) [5 points] You are supposed to copy files from my directory ~piatnitskaiag/CS2750/CProgr/Ch8. What permissions do you need for all directories on this path?

*See p. 171.*

(b) [5 points] What permissions for a directory do you need in order to be able to create and delete files in this directory?

*See p. 171.*

1. [5 points] Explain the difference between two types of special files.

*See p. 168.*

1. [10 points] Write a command to do each of the following:
2. There is a file Data located in your grandparent directory. In the current directory create a hard link named DataHard to this file.

ln ../../Data DataHard

1. Now a file Data is located in your parent directory. Create a soft link named DataSoft in the current directory to this file.

ln -s ../Data DataHard

1. [10 points] Write a command that will find and display names of all files that include a substring char, under your parent directory.

find .. -name char -print

1. [15 points] (a) Explain what kind of information is stored in a directory entry for each file.

*See p. 167.*

(b) Explain what kind of information is stored in an *i*-node.

*See p. 176.*

1. Explain the connection between concepts of *i*-node, *i*-number and *i*-list.

*See p. 176 – 177.*

1. [12 points] Indicate if each of the following character arrays implements a C-style string; if yes, indicate a string length.

|  |  |  |
| --- | --- | --- |
| Array of characters | Implements a string? (Yes/No) | String length |
| char \* a = {‘a’, ‘b’, ‘\0’, ‘1’, ‘5’} | Yes | 2 |
| char \* b = {‘H’, ‘E’, ‘L’ ‘L’, ‘O’, ‘\n’} | No | X |
| char \* a = {‘w’, ‘z’, ‘\n’, ‘x’, ‘t’} | No | X |
| char \* a = {‘r’, ‘u’, ‘m’, ‘y’, ‘k’, ‘\0’} | Yes | 5 |

1. [12 points] The command to run a program is: **PROGR one two three**

In the source file you have the following line:

**printf(“%c, %c, %c\n”, argv[2][0], argv[3][1], argv[1][2]);**

Show exactly what will be printed:

t, h, e

1. [22 points] In the following C statements you are supposed to find 11 errors (there may be more than one error per statement); identify mistakes and write down corrected versions:

Note that wrong symbols are highlighted in yellow; symbols that were missed are typed in red.

a) Printf("You entered the value: %d\n", &number);

printf(“You entered the value: %d\n”, number);

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

b) Scanf("%d", **&**integer)**;**

scanf("%d", &integer);

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

c) print**f**("The sum of %d and %d is equal to %d\n", v, u**, v+u**);

printf("The sum of %d and %d is equal to %d\n", v, u, v+u);

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

1. d) printf("The product is %d"\n**”**, a, b);

printf("The product is %d"\n”, a);

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

e) scan**f**(“%f**”**, &value);

scanf(“%f”, &value);

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

f) putchar(“ **‘**\n” **‘**);

putchar(‘\n‘);

1. [20 points] Write a definition of a C function replace that has a string (a pointer to a character) as a parameter and replaces each occurrence of a character A in this string with a character X. It has the following prototype: char \* replace (char \* str);

You are NOT allowed to use a function **strlen** from **string** library.

char \* replace (char \* str){

int k;

for(k = 0; str[k] != ‘\0’; k++)

{

if (str[k] == ‘A’)

str[k] = ‘X’;

}

return str;

}

1. [15 points] There are several errors in the code of the C function shown in left column of the table below. Identify them and write the corrected code in the right column of the table.

|  |  |
| --- | --- |
| int count\_spaces (const char \* filename)  { FILE fp;  int n = 0;  if ((fp = fopen(filename, “r”)) == NULL) {  while (fgetc(fp) != EOF)  if(fgetc(fp) = “ “)  n++;  fclose(filename);  } | Corrected function:  int count\_spaces (const char \* filename)  { FILE \* fp;  int n = 0;  char ch; /\* new variable \*/  if ((fp = fopen(filename, “r”)) != NULL) {  while ((ch = fgetc(fp)) != EOF)  if(ch == ‘ ‘)  n++;  fclose(fp);  return n;  } |

1. [16 points] Write a C function with the following prototype:

*int \* create\_arr(int n, int initial\_value)*;

The function should return a pointer to a dynamically allocated array with n members, each of which is initialized to initial\_value. The return value should be NULL if the array can’t be allocated.

int \* create\_arr(int n, int initial\_value){

int k;

int \* arr = malloc(n \* sizeof(int));

if (arr == NULL)

return NULL;

for(k = 0; k <= n; k++)

arr[k] = initial\_value;

return arr;

}

1. (a) [5 points] Suppose that the macro Mhas been defined as follows: #define M 0

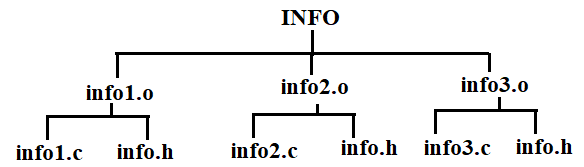
Indicate whether each of the following tests will produce true or false value.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| test | #if M | #ifdef M | #ifndef M | #if defined M | #if !defined M |
| true/false | false | true | false | true | false |

(b) [8 points] Define a macro DIFF that has two parameters and will be replaced with the difference of their squares when appears in the code.

#define DIFF(x, y) (((x) \* (x)) – ((y) \* (y)))

1. [10 points] Write a makefile according to a dependency tree shown below.



INFO: info1.o info2.o info3.o

gcc -o INFO info\*.o

info1.o: info1.c info.h

gcc -c info1.c

info2.o: info2.c info.h

gcc -c info2.c

info3.o: info3.c info.h

gcc -c info3.c

*Problems for extra points*:

1. [10 points] Use properties of scanf function to write a code fragment that will read a 4-digit integer number from a keyboard and find a sum of its digits.

int sum;

scanf(“%1d%1d%1d%1d”, &n1, &n2, &n3, &n4)

sum = n1 + n2 + n3 + n4;

1. [10 points] Suppose you have multiple files in your program, and functions’ prototypes are placed in the file Header.h. Explain why it is necessary to enclose all statements in this file inside the construction like the following:

#ifndef HEADER

#define HEADER

/\* information from Header.h file \*/

#endif

This prevents a header file from being included several times, which would be a mistake.