**CS2750 System programming and Tools Fall 2018**

**KEY TO** **TEST #1 [total 220 points out of 200]**

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

*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. [12 points] Describe the structure of Linux showing different components from hardware to the user.

For the diagram and explanations see pp. 5 – 6.

1. [11 points] Describe at least one way to find documentation for a command that is a built-in Bash command.

For example, you may type: **help**

(typing **man command** is a universal way to get information about any **command**, not necessarily built-in)

|  |  |
| --- | --- |
| (a)[6 points] Show the output of the following sequence of commands:  var=”Hello”  echo $var  (var=”Hi”; echo $var); echo $var  *Output:*  Hello  Hi  Hello | (b)[8 points] Show the output of the following sequence of commands:  var=”Hi”  export var  echo $var  (echo $var; var=”Hello”; echo $var); echo $var  *Output:*  Hi  Hi  Hello  Hi |

Explanation: if commands are enclosed into parentheses, they are executed as a unit in a new child shell. If a variable is defined in a parent shell, it is not transferred to a child shell unless a command **export** is used.

1. [12 points] How will you make sure that your customization of Bash shell does not go away when you log out of the system?

Put your customization either in **.bashrc** or **bash\_profile** file.

1. (a) [5 points] Write down default permissions (for zero umask value) in octal format:

|  |  |
| --- | --- |
| for a regular file:  666 | for a directory:  777 |

1. [5 points] What umask value will change default permissions (from part (a)) for a regular file to:

rw-r----- ?

Since rw-r----- corresponds to the octal value 640, umask is 026

(c) [6 points] Assuming that umask now has the value from the part (b), show permissions for a newly created directory:

Applying umask value 026 to the default directory permissions 777: 777 – 026 = 751

|  |  |
| --- | --- |
| rwx triplets:  rwxr-x—x | Octal:  751 |

1. [25 points]

|  |  |
| --- | --- |
| Given a script problem.sh:  #!/bin/bash  echo ‘$0 = ‘ $0  echo ‘$2 = ‘ $2  echo ‘$# = ‘ $#  shift 3  echo ‘$2 = ‘ $2  echo ‘$# = ‘ $#  echo ‘$\* = ‘ $\* | Show what will be displayed when this script is invoked in the following way:  bash problem.sh 1 2 3 4 5  **------------------------------**  *OUTPUT:*  $0 = problem.sh  $2 = 2  $# = 5  $2 = 5  $# = 2  $\* = 4 5 |

1. [15 points] You want to check if a file **info** contains a pattern **first**. However, you don’t want to see error messages or even lines that have this pattern. All you want to see on a screen is either

**info contains first**

or

**info does not contain first**

Write a code fragment that will solve this problem.

**if** grep -q first info

**then**

echo info contains first

**else**

echo info does not contain first

**fi**

1. [20 points] Rewrite the following fragment using a **case** command:

|  |  |
| --- | --- |
| **if** [[ $str == “one” ]]  **then**  echo 1  **elif** [[ $str == “two” ]]  **then**  echo 2  **elif** [[ $str == “five” ]]  **then**  echo 5  **elif** [[ $str == “ten” ]]  **then**  echo 10  **else**  echo 0  **fi** | Alternative syntax with **case**:  case $str in  “one”)  echo 1  ;;  “two”)  echo 2  ;;  “five”)  echo 5  ;;  “ten”)  echo 10  ;;  \*)  echo 0  **esac** |

1. [25 points] Write a bash script that is supposed to have one positional parameter, which is a directory name. This script should: (a) check if a number and a type of positional parameters are correct; if not, exit with a proper status; (b) switch to the directory, the name of which is provided as a positional parameter; (c) for each file in this directory check if this file is executable by a user;

(d) display either YES or NO based on the outcome of a checking procedure in step (c).

#!/bin/bash

**if** [[ $# != 1 ]]

**then**

echo wrong number of parameters

exit 1

**fi**

**if** [[ -d $1 ]]

**then**

cd $1

**for** file in \*

**do**

**if** [[ -x $file ]]

**then**

echo YES

**else**

echo NO

**fi**

**done**

**else**

echo wrong parameter type

exit 2

**fi**

1. [25 points] Write a code fragment that will take several strings from the user and display them together separated with spaces in the end of execution. Use a while or until loop, on each iteration of which you take one string from a user and concatenate it with the currently existing string. Perform initialization before starting a loop.

|  |  |
| --- | --- |
| Without using arrays:  #!/bin/bash  result=""  read new  **while** [[ "$new" != "no" ]]  **do**  result="$result $new"  echo enter  read new  **done**  echo $result  Also, it is possible to use **+** to concatenate strings | Using arrays:  #!/bin/bash  result=()  read new  **while** [[ "$new" != "no" ]]  **do**  result+=(" $new")  echo enter  read new  **done**  echo "${result[@]}" |

1. [15 points] In a file named Cars each line has the format:

make,model,year,miles,price

(note that a field separator is different from a default space). Write a single Linux command (you may use input/output redirection and piping) that will do the following: (a) replace all upper case letters with lower case letters, (b) sort this file by models in reverse alphabetical order, (c) replace all commas with columns, (d) put the result into the file Result.txt.

tr A-Z a-z < Cars | sort -rt, --key=2 | tr , : > Result.txt

1. [15 points]

|  |  |
| --- | --- |
| Given a script result.sh:  #!/bin/bash  **function** cmp()  { local s1=”$1”  local s2=”$2”  **if** [[ $s1 == $s2 ]]  **then** echo EQUAL  **elif** [[ $s1 > $s2 ]]  **then** echo echo $1  else echo $2  **fi**  }  cmp $1 $2 | Show what will be displayed when this script is invoked in the following way:  bash result.sh LINUX UNIX  **------------------------------**  *OUTPUT:*  UNIX  -----------------------------  Explanation: LINUX comes first in lexicographic order, since L < U, so UNIX > LINUX |

1. [15 points] The file Text is shown in the left column of the table below. Show the strings from this file that will match each of the regular expressions.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| $$$aaa  bcd$  $  abc  abcd$  a$bc | |  |  |  | | --- | --- | --- | | Regular expression | Matching string(s) | Explanation | | [^\$]$ | $$$aaa  abc  a$bc | The last symbol in the row is different from $ | | ^[^\$][bcd]\* \$ | bcd$  abcd$  a$bc | The first symbol in the line is different from $; it is followed by 0 or more symbols from [bcd], followed by $ | | \$[a-c]\* | $$$aaa  bcd$  $  abcd$  a$bc | The symbol $ (any position in the line) is followed by 0 or more symbols from [a-c] | |