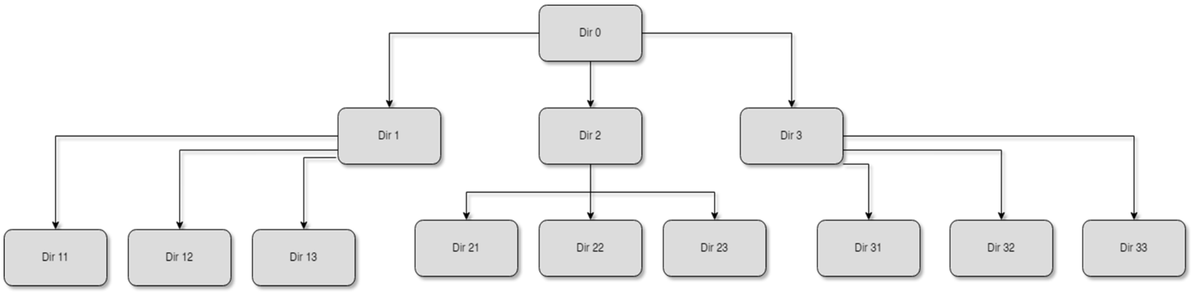
CSE 4502 : Operating Systems Lab
Islamic University of Technology (IUT)
Department of CSE
Time : 1 Hour 20 Mins
Date: Monday, 09 November 2022
Total Marks: 50

1. Write a bash script having the name `crdir.sh` with an aim to create multiple directories within directories. It will take 2 parameters; the *first* one represents the number of levels there must be in the hierarchy, while the *second* one represents the number of child directories that each directory in each of the levels must have. For example: 20

```
$./crdir.sh 2 3
```

The first parameter represents that there will be 2 levels of child directories. And the second parameter represents that each directory in each of the levels will have 3 child directories. See the following *figure* for better comprehension.



You must ensure that the script creates the root directory (`Dir 0` in *figure*) under the `Documents` directory. Also, if the user makes any usage error while typing the command, the following provisions must be there,

a. While executing the script without the parameters, syntax error should be raised and must be addressed with a prompt of the following nature,

```
$./crdir.sh

Syntax Error!!
Syntax : ./crdir.sh <parameter_1> <parameter_2>
For more information type: ./crdir.sh -h
```

b. When the script is called with `-h`, the output should be the following.

```
$./crdir.sh -h

-./crdir.sh creates multiple directories within directories in the
'Documents' directory.
- Syntax : ./crdir.sh <parameter_1> <parameter_2>
-- Parameter 1 <parameter_1> : Number of levels in the hierarchy.
-- Parameter 2 <parameter_2> : Number of child directories in each
directory of each level.
```

2. A common problem in shell script is that it has no provision for floating point numbers. Write a shell script having the name `floater.sh` that takes *three* numbers as parameters, where the *first* number is not divisible by the *second* and generates a string representation of the result in decimals up to *n*-significant digits, where *n* is the *third* parameter of the script. It should support the following requirements. 15

You must ensure that if the user makes any usage error while typing the command, the following provisions must be there,

a. While executing the script without the parameters, syntax error should be raised and must be addressed with a prompt of the following nature,

```
$./floater.sh

Syntax Error!!
Syntax : ./floater.sh <number_1> <number_2> <number_3>
For more information type: ./floater.sh -h
```

b. When the script is called with `-h`, the output should be the following.

```
$./floater.sh -h

-./floater.sh prints the string representation of the floating-point
quotient of two numbers that are not divisible by one another, up to
n-significant digits.
- Syntax : ./floater.sh <number_1> <number_2> <number_3>
-- Parameter 1 <number_1> : Dividend.
-- Parameter 2 <number_2> : Divisor (must be greater than zero).
-- Parameter 3 <number_3> : Number of significant digits after the
decimal point.
```

c. You should ensure that the *n*-th digit of the quotient is rounded up if the subsequent digit is greater than 5.

d. You should check for division by zero error.

The input and output should have the following format:

Input	Output
`$./floater.sh 2 3 4`	`1.6667`
`$./floater.sh 5 3 3`	`2.667`

`$./floater.sh 5 0 3`	`Division by zero Error!!`
	`For more information type: ./floater.sh -h`

3. Write a shell script that will prompt the user for an input string. The script will return an output string only consisting of the **vowels** from the given input string. 10

Note: strings will not contain any *spaces*.

The input and output should have the following format:

Input	Output
$./vowels.sh maurisluctuserosatnibhiaculistempus	auiuueiaiiauieu

4. Write a shell script fibonacci_bros.sh that prints the Fibonacci Series up to *n*-th digit. The script should accept the value of *n* as an argument. 5

The input and output should have the following format:

Input	Output
$./ fibonacci_bros.sh 5	The first 5 digits of the Fibonacci series are: 1 1 2 3 5