**ASSIGNMENT01**

**OBSERVATION TABLE**

**NAME:HARSHA TIWARI**

**PRN:240340126004**

**QUESTION01\_OBSERVATION\_TABLE\_Q1.sh**

1. The next line echo =enter the file name\_\_displays the message "Enter file name: " to the user.
2. The line read filename prompts the user to enter a file name, which is then stored in the variable filename.
3. The line if $filename then checks if the file specified by the user exists. The-foption checks if the file is a regular file (not a directory or a device file).
4. If the file exists, the line cat "$filename"displays the contents of the file.
5. If the file does not exist, the line echo $filename \*does not exist”displays the message "'<filename>' does not exist".
6. The script then repeats the same process again, prompting the user to enter another file name and checking if it exists.

**QUESTION02\_OBSERVATION\_TABLE\_Q2.sh**

* 1. he string "Enter number: " to the console, prompting the user to enter a number.
  2. read num: This command reads a value from the standard input (the keyboard) and stores it in the variable num.
  3. if [ $(($num%2)) -eq 0 ]; then: This is a conditional statement that checks if the remainder of the division of num by 2 is equal to 0. If it is, then the number is even.
     1. $(($num%2)): This is an arithmetic expansion that calculates the remainder of the division of num by 2.
  4. 0: This is the value that $(($num%2))is compared against.

1. echo "'$num' is even": If the condition in the if statement is true, this command prints the string "'$num' is even" to the console.
2. else: If the condition in the if statement is false, the commands in the else block are executed.
3. echo "'$num' is odd": If the condition in the if statement is false, this command prints the string "'$num' is odd" to the console.
4. fi: This keyword marks the end of the if-else statement.

**QUESTION03\_OBSERVATION\_TABLE\_Q3.sh**

**The first column shows the iteration number (starting from 0).**

1. The second column shows the value of num at each iteration.
2. The third column shows the value of fact at each iteration.
3. At each iteration, the script multiplies fact by num and decrements num by 1. The loop continues until num is 1.
4. For example, if the user enters 5, the script will calculate the factorial as follows:

5!=5\*4\*3\*2\*1=120

4!=4\*3\*2\*1=24

The final result is fact = 120, which is the factorial of 5.

**QUESTION04\_OBSERVATION\_TABLE\_Q4.sh**

| **Column Name** | **Description** |
| --- | --- |
| File Name | The name of the file before the renaming process |
| File Type | The file extension before the renaming process (in this case, ".txt") |
| Command | The command used to rename the file (in this case, the mv \command with the sed command to change the file extension) |
| New File Name | The name of the file after the renaming process |
| New File Type | The file extension after the renaming process (in this case, ".bak") |

**QUESTION05\_OBSERVATION\_TABLE\_Q5.sh**

The script prompts the user to enter a directory path.

1. The find command is used to search for files and directories recursively within the specified directory.
2. The -type f option tells find to only consider files, and the-type d option tells it to only consider directories.
3. The wc -l command is used to count the number of lines output by , find which corresponds to the number of files or directories found.
4. The results are stored in the filecount and dircount variables, respectively.
5. Finally, the script prints the file count and directory count to the console.

**QUESTION06\_OBSERVATION\_TABLE\_Q6.sh**

1. The script first prompts the user to enter the name of the file they would like to read. It then checks if the file exists using the -f test. If the file does not exist, the script prints an error message and exits.
2. If the file does exist, the script uses the cat command to print the contents of the file, and pipes the output to the revcommand, which reverses the order of each line.
3. The script then repeats this process for a second file, using the same code as before.

**QUESTION07\_OBSERVATION\_TABLE\_Q7.sh**

1. It initializes a variable reverse to store the reverse of the number.It stores the original number in a separate var num1.
2. It enters a loop that continues until the number becomes 0.
3. In each iteration of the loop, it gets the remainder of the number divided by 10, appends it to the reverse number, and removes the last digit of the number by dividing it by 10.
4. After the loop, it compares the reverse number with the original number.
5. If they are equal, it prints that the number is a palindrome. Otherwise, it prints that the number is not a palindrome.

**QUESTION08\_OBSERVATION\_TABLE\_Q8.sh**

1. It then checks if the directory exists using the -d flag. If the directory does not exist, it will print "No file update in 24 hours".
2. If the directory does exist, it uses the find command to search for files (-type f) that have been modified in the last 24 hours (-mtime -1). The -ls flag is used to display the files in a list format.

**QUESTION09\_OBSERVATION\_TABLE\_Q9.sh**

1. the read command is used to read the user's input and store it in a variable called pattern.
2. The find command is used to search for files in the current directory and its subdirectories. The type f flag tells find to only search for files (not directories).
3. The |character is used to pipe the output of the find command to the ls command. This means that the output of find (a list of files) is used as the input to ls .
4. The ls command is used to list the files found by find to-l flag tells ls to print detailed information about each file.

**QUESTION10\_OBSERVATION\_TABLE\_Q10.sh**

1. read var Read user input and store it in the variable var.
2. **if** [ -d "$var" ]; then- Check if the input is a directory.
3. **echo** "It is directory"- If the input is a directory, print "It is directory".
4. **elif** [ -f "$var" ]; then - Check if the input is a regular file.
5. **echo** "It is a file"- If the input is a regular file, print "It is a file".
6. **else**- If the input is neither a directory nor a regular file, execute the code in this block.
7. **echo** "something else" - Print "something else" if the input is neither a directory nor a regular file.
8. fi- End of the if statement.