Week 2 - Assignment - Answer Explanations - Q1.1

Q1.1) Breakdown

```
progress_bar(){
    local count=0
    percent=$1
    limit=$(expr $percent / 2)
    while [ $count -le $limit ]
    do
        echo -ne "+"
        let count=count+1
    done
    while [ $count -le 50 ]
        do
            echo -ne '.'
            let count=count+1
        done
        echo -ne "    [ $percent % ]\r"
    #echo -ne "\n"
```

Here, I am defining a function progress_bar() to create a live progress-bar in the terminal.

- I am taking the total percentage (percent) to draw (say 46%, then the bar would be drawn till 45%) as parameter.
- I have taken the length of the bar as 50 characters, hence, I am dividing the percent by 2 and saving the resulting the expression in limit (using expr $percent / 2, which gives an integer value)
- First while loop -
    - Say, percent = 45 => limit = 22. I am iterating till the local var. count <= limit and while iterating, I am printing the character '+' (for the progress)

- Second while loop -
    - To create a nice effect, I am filling the rest of the bar (which doesn't have progress) with '.'. (So, here, 100-22 = 78% of the bar would be of '.')
- echo -ne " [$percent %]\r" here, I am doing carriage return \r so that, every the time the loop iterates and count gets updated, the bar also updates by increasing the no. of '+'s

Week 2 - Assignment - Answer Explanations - Q1.1

```
#start of file manager tasks
(echo './' > dot.txt) #task1
percent=2
progress_bar "$percent"
sleep 1
((tr "[/]" "[&]" < path.txt) > pathi.txt) #task2
let percent=percent+6
progress_bar "$percent"
sleep 1
```

- This is the start of my old code (week 1). For each of the commands in this code, I am calling the progress_bar function along with the percent parameter (that I am updating as go from one command to the next)
- That's all.

```
rsgr@GokulsPavilion:~/lukog/week2$ ls -l
total 20
-rw-r--r-- 1 rsgr rsgr 2213 Jun 10 23:52 ascii-art.sh
-rwxr-xr-x 1 rsgr rsgr 2225 Jun 10 19:12 basic-file.sh
-rw-r--r-- 1 rsgr rsgr   80 Jun 11 00:26 path.txt
-rwxr-xr-x 1 rsgr rsgr 3538 Jun 10 19:13 q1.2.sh
-rw-r--r-- 1 rsgr rsgr 2805 Jun 10 23:55 the-valor.sh
rsgr@GokulsPavilion:~/lukog/week2$
```

```
rsgr@GokulsPavilion:~/lukog/week2$ bash basic-file.sh
Progress made so far:
--------------------------------------------------
+++++.....................................    [ 8 % ]
```

```
rsgr@GokulsPavilion:~/lukog/week2$ bash basic-file.sh
Progress made so far:
--------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++.....    [ 90 % ]
```

Week 2 - Assignment - Answer Explanations - Q1.1

```
rsgr@GokulsPavilion:~/lukog/week2$ bash basic-file.sh
Progress made so far:
--------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++    [ 100 % ]
--------------------------------------------------
All tasks completed.
```

```
rsgr@GokulsPavilion:~/lukog/week2$ ls -l
total 32
-rw-r--r-- 1 rsgr rsgr 2213 Jun 10 23:52 ascii-art.sh
-rwxr-xr-x 1 rsgr rsgr 2225 Jun 10 19:12 basic-file.sh
drwxr-xr-x 2 rsgr rsgr 4096 Jun 11 00:28 end
drwxr-xr-x 2 rsgr rsgr 4096 Jun 11 00:28 ends
-rw-r--r-- 1 rsgr rsgr   80 Jun 11 00:26 path.txt
-rwxr-xr-x 1 rsgr rsgr 3538 Jun 10 19:13 q1.2.sh
-rw-r--r-- 1 rsgr rsgr 2805 Jun 10 23:55 the-valor.sh
drwxr-xr-x 3 rsgr rsgr 4096 Jun 11 00:27 this
```

Week 2 - Assignment - Answer Explanations - Q1.2

Q 1.2)

```
#Gokularamanan R S, 23B1854
alphabet=$1
if [ $alphabet = 'a' ] || [ $alphabet = 'A' ]
then
   echo   -e   "    /\ \n   /__\ \n  /----\ \n /      \ "
   #echo     "   /__\ "
   #echo  -E  "  /----\ "
   #echo  -E " /      \ "
elif  [ $alphabet = 'b' ] || [ $alphabet = 'B' ]
then
    echo -e " ______ \n||___|| \n||    || \n||____||  "
    #echo "||___|| "
    #echo ""
    #echo "||____||"
elif [ $alphabet = 'c' ] || [ $alphabet = 'C' ]
then
    echo " _____  "
    echo "||     "
    echo "||     "
    echo "||____  "
elif [ $alphabet = 'd' ] || [ $alphabet = 'D' ]
then
    echo " ______ "
    echo "  |   ] "
    echo "  |   ] "
    echo " _|___] "
```

- Here, I am using an if-elif-else conditional construct to check what alphabet (irrespective of the case - I have used || - or case to do it) the user has given and after checking what the alphabet it is, I am printing the special ascii art for it.
- NOTE : It can be seen that for ‘a’ and ‘b’, I have written the echo statement in one line, but for others, in multiple echo statements. I did this change for ‘a’ and ‘b’ only for testing it for q1.3 (but later in q1.3, I totally changed my approach to arrays)

Week 2 - Assignment - Answer Explanations - Q1.2

- Output for some inputs

```
rsgr@GokulsPavilion:~/lukog/week2$ bash q1.2.sh d
 _____
 |   ]
 |   ]
 |___]
rsgr@GokulsPavilion:~/lukog/week2$ bash q1.2.sh c
 _____
||
||
||____
rsgr@GokulsPavilion:~/lukog/week2$ bash q1.2.sh f
 ######
 #
 #===
 #
rsgr@GokulsPavilion:~/lukog/week2$ bash q1.2.sh i
---ii---
   ii
   ii
---ii---
```

```
rsgr@GokulsPavilion:~/lukog/week2$ bash q1.2.sh r
 ____
||_)
||\
|| \
rsgr@GokulsPavilion:~/lukog/week2$ bash q1.2.sh v
\\    //
 \\  //
  \\//
   VV
rsgr@GokulsPavilion:~/lukog/week2$ bash q1.2.sh x
\\   //
 \\ //
 // \\
//   \\
```

Week 2 - Assignment - Answer Explanations - Q1.3

```
st_line(){
for ((i=0;i<2;i++))
do
    echo -e "================================================
done
}
a=("    /\ " " "   /__\ " "  /----\ " " /      \ ")
b=("  _______" "  ||___||" " ||    ||" "||____||")
c=(" _____ " "||     " "||     " "||____ ")
d=(" ______ " "  |   ]" "  |   ]" " _|___]")
e=(" ______" " {     " " {===  " " {___  ")
f=(" ######" " #     " " #===  " " #     ")
g=("   _____" "  /     " " /  ==>>" "/_____/ ")
h=(" [     ]" " [=====]" " [=====]" " [     ]")
i=(" ==ii== " "   ii   " "   ii   " " ==ii== ")
j=("=====" "   jj" "   jj" " \_j}")
k=("k //  " "k//   " "k\\\\\\   " "k \\\\\\  ")
l=("##   " "##   " "##___" "#####")
m=("__    __" "||\  /||" "|| \/ ||" "|| -- ||")
n=("__   __" "||\\\  ||" "|| \\\ ||" "||  \\\||")
o=("=======" "=  o  =" "=  o  =" "=======")
```

```
p=("PPppp" "Pp__}" "Pp   " "p    ")
q=("====|||  " "====|||  " "==  q\\\\\\  " "======\\\\\\ ")
r=("====" "||_)" "||\\\\ " "|| \\\\")
s=("______" "||    " "||___ " "____||")
t=("______" "  ||  " "  ||  " "  ||  ")
u=("U    U" "U    U" "U    U" " UUUU ")
v=("\\\\\\    //" " \\\\\\  // " "  \\\\\\//  " "   VV   ")
w=("w  w  w" "w  w  w" "w  w  w" "ww/ \ww")
x=("\\\\\\   // " " \\\\\\ //  " " // \\\\\\  " "//   \\\\\\ ")
y=("\\\\\\  //" " \\\\\\// " "  ||  " "  ||  ")
z=("=====  " "   //  " "  //   " " //___")
#Now, it starts
for ((i=0;i<=3;i++))
do
    echo -e "${w[$i]} ${e[$i]} ${l[$i]} ${c[$i]} ${o[$i]} ${m[$i]} ${e[$i]}\n"
done
```

Week 2 - Assignment - Answer Explanations - Q1.3

- As seen above, I changed the ascii art for each character to an array from last question (q1.2)
    - For ‘a’, the array a contains its ascii art, for ‘b’, the array ‘b’ contains its ascii art, and so on.
    - Each of the arrays have length equal to 4 => There are 4 strings in each array
    - I have done the above changes so as to print the art in horizontal direction (which is not possible to do if I used the logic of q1.2 as such)
    - Now, iterating through a for loop 4 times, each time, I am printing the ith element of each letter separated by a space.
    - So, first iteration (i=0), w[0] e[0] l[0] c[0] m[0] e[0] gets printed
    - Therefore, towards the end of the loop, the following would have been printed :

        w[0] e[0] l[0] c[0] m[0] e[0]
        w[1] e[1] l[1] c[1] m[1] e[1]
        w[2] e[2] l[2] c[2] m[2] e[2]
        w[3] e[3] l[3] c[3] m[3] e[3]

Thus, the complete word has been printed. Similarly, we can also print for any given sentence.

```
rsgr@GokulsPavilion:~/lukog/week2$ bash ascii-art.sh
 w   w   w   ______  ##      _____   =======  __     __  ______
 w   w   w  {       ##     ||     =   o   =  ||\   /||  {
 w   w   w  {===    ##___  ||     =   o   =  || \/ ||  {===
 ww/ \ww   {___    #####  ||____  =======  ||  --  ||  {___
rsgr@GokulsPavilion:~/lukog/week2$
```

(WELCOME is printed, in a fancy font style (my own creation))