STUDENT'S NAME Anup Doddannavar		TOTAL MARKS OBTAINED
CLASS A	SUBJECT LSS	
ROLL NO. 14	DATE 31/3/22	

Day 2

Programming Constructors.

1) The shbang line
→ It is the very first line of the script and kernel knows what shell will be interpreting the lines in the script
It consists of #! followed by the full path name.
eg:- #! /bin/bash

2) Comments
→ These are descriptive material preceded by #
# comment.

3) Local variables.
→ local variables are in scope for the current shell when a script ends they are no longer available
These are defined with built-in declare function

4) Global variables.
→ These are called as environment variables, and are created with export built-in function.
They are set for currently running shell. Built-in function with -x option also sets an environment variable.

eg:- export Variable name = value
declare -x Variablename = value
export path= /bin:/usr/bin:

5. Extracting values from variables

→ To extract a value from variable a $ is used.

eg:- echo $ name, path, etc

Q2. Looping and Conditional Statements

1) if Statement.
This block will process if specified condition is true

eg:-
```
a=10
b=20
if [$a == $b]
then
echo " a equal to b"
fi.
```

2) if else. Statement
This if -specified condition is not true in if part then else part will be executed.

eg:-
```
a=20
b=20
if [$ a == $b]
then
    echo "a equal to b"
else
    echo "a is not equal to b"
fi
```

3) Switch Statement

→ Case Statment works of a switch statement if specified value match with the pattern then it will execute a block of that particular pattern.

Scanned by TapScanner

STUDENT'S NAME	Anup Doddannavar		TOTAL MARKS OBTAINED
CLASS	A	SUBJECT LSS	
ROLL NO.	14	DATE 31/3/22	

eg:-
```
Cars = "BMW"

case "$cars" in
     # case 1.
     "mercedes" ) echo " Headquaters - Germany";;
     # case 2
     "bmw") echo " Headquaters - Italy";;
```

4) Looping Statements.

(a) While Statement: Command is evaluated and based on the result loop will executed.

eg:-
```
      a=0
while [ $a -lt 10]
 do
    echo $a
    a=`expr $a + 1`
```

(b) for Statements :- for loop operates on lists of items. It repeats a set of commands for every item

eg:
```
  for a in 1 2 3 4 5
   do
     if [ $a == 5 ]
     then
         break;
     fi
     echo " Iteration on $a"
     done
```

(c) Until Statement: It is executed many times when the condition / command evolves to false

Choudhary Gold

eg:
```
a=0
until [$a  -gt 10]
do
    echo $a
    a= 'expr $a + 1`
done.
```