

# LINUX – Intro into Scripts

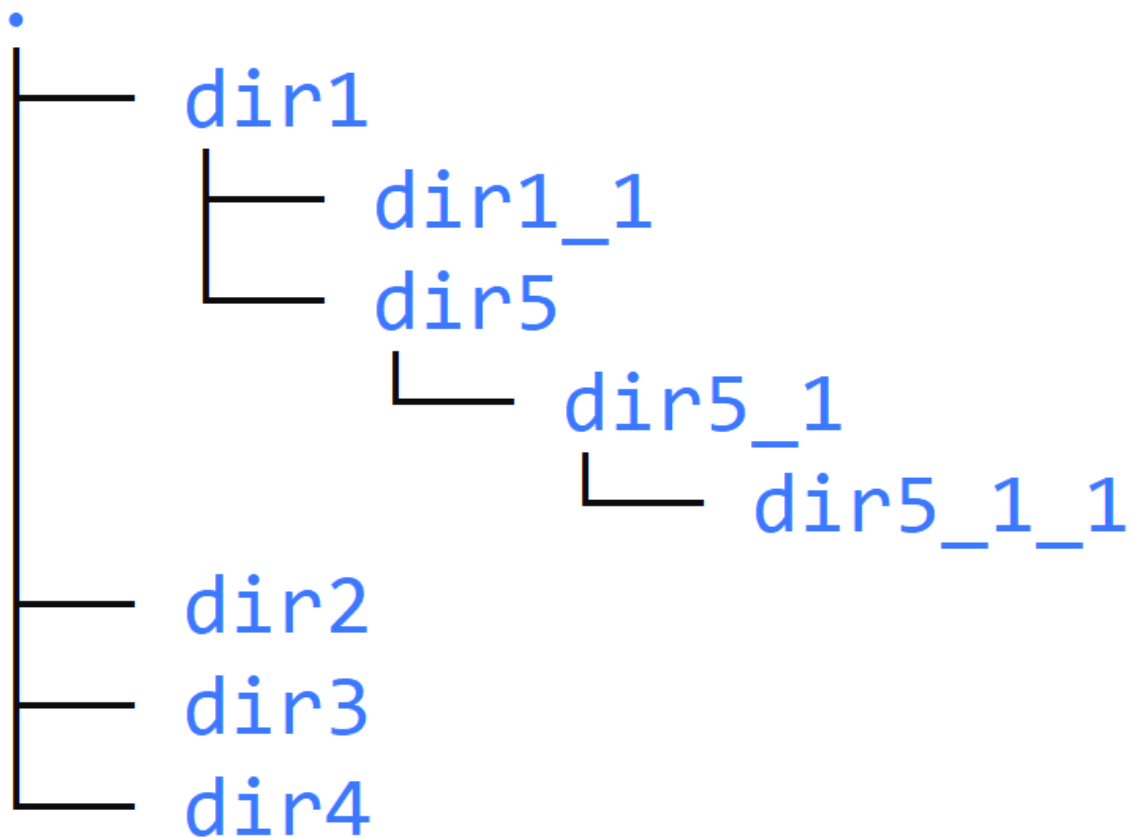
**סקריפט** הינו אוסף פקודות של מעטפת.

על ידי הרצה של סקריפט ניתנן להריץ באופן סידרתי או היררכי את הפקודות שכלולות בתוכו.

כעת בעזרת תרגול קטן נראה מהי התועלת של סקריפטים. תחילה נבצע כמה פעולות בשורת הפקודה ואז נהפוך אותן לסקריפט להרצה סדרתית.

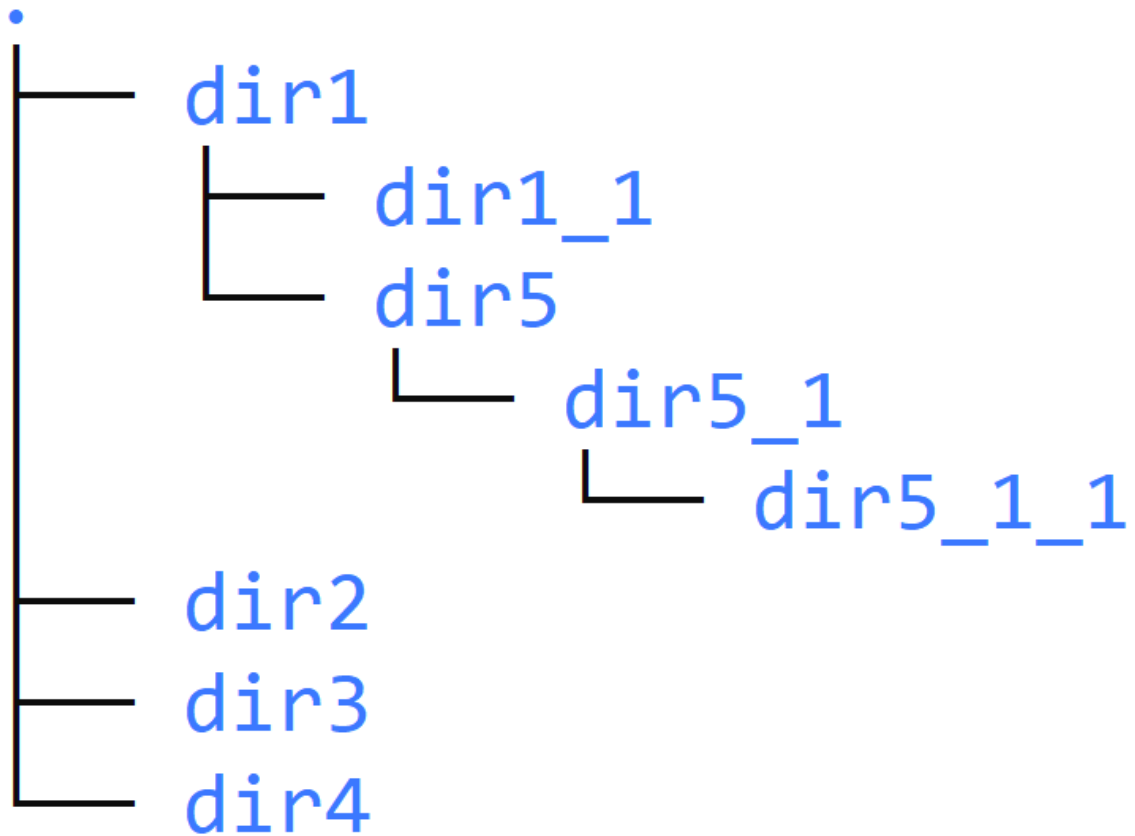
# LINUX – Intro into Scripts

בנו עץ ספריות כמו בדוגמא :



# LINUX – Intro into Scripts

ניתן לבנות אותו על ידי פקודות האלה :



```
mkdir dir1 dir2 dir3 dir4
```

```
cd dir1
```

```
mkdir dir1_1
```

```
mkdir -p dir5/dir5_1/dir5_1_1
```

# LINUX – Intro into Scripts

כעת נמחק את תוכן התיקיה  
הראשית ונוודא שהצלחנו:

```
teacher@DESKTOP-CT14MRR:~$ rm -r *  
teacher@DESKTOP-CT14MRR:~$ ls -l  
total 0
```

# LINUX – Intro into Scripts

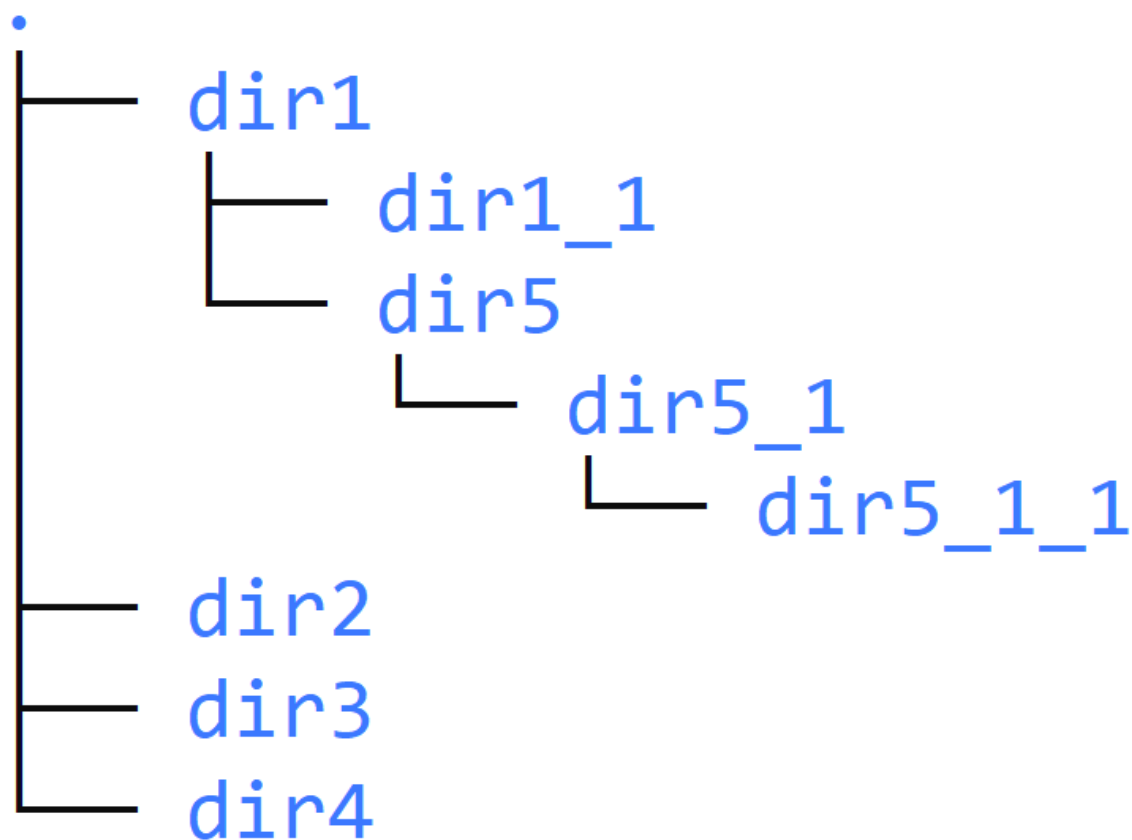
נהפוך את הפקודות שכתבנו לסקריפט בשם:

**00\_A\_DIR\_TREE.sh**

נשתמש בעורך nano:

**nano 00\_A\_DIR\_TREE.sh**

תוכן הסקריפט :



```
#!/bin/bash
mkdir dir1 dir2 dir3 dir4
cd dir1
mkdir dir1_1
mkdir -p dir5/dir5_1/dir5_1_1
```

# LINUX – Intro into Scripts

ההוראות לשינוי ההרשאות והרצת הסקריפט :

```
chmod +x 00_A_DIR_TREE.sh
```

```
./00_A_DIR_TREE.sh
```

# LINUX – Intro into Scripts

כעת נתבונן בכמה

**דוגמאות של סקריפטים**

שמציגים יכולות ואפשרויות

שונות ומגוונות

# LINUX – Intro into Scripts

כך נוכל לקבל קלט מהמשתמש :

01\_ReadUserInput.sh

-----

```
#!/bin/bash
```

```
echo "Enter your input:"
```

```
read var
```

```
echo "You entered :$var"
```



# LINUX – Intro into Scripts

את שם התיקיות נקבל כקלט מהמשתמש :

```
#!/bin/bash
```

```
#02_CustomMK_4DIR_and_PATH.sh
```

```
echo "Enter Directory Name:"
```

```
read dir
```

```
mkdir ${dir}_1 ${dir}_2 ${dir}_3 ${dir}_4
```

```
mkdir -p ${dir}_1/${dir}_2/${dir}_3/${dir}_4
```

# LINUX – Intro into Scripts

```
$ ./02_Custom_MK_4DIRs_and_PATH.sh
```

Enter Directory Name:

stam

```
$ tree
```

```
|— stam_1
|   └─ stam_2
|       └─ stam_3
|           └─ stam_4
|— stam_2
|— stam_3
|— stam_4
```

## Tip

In order to remove all the directories we just created use command **rm**

```
$ rm -R stam*
```

# LINUX – Intro into Scripts

לולאות בתוך הסקריפטים (1) :

```
for i in 1 2 3 4
do
    echo $i
done
```

# LINUX – Intro into Scripts

לולאות בתוך הסקריפטים (2) :

```
for i in {5..8}
```

```
do
```

```
    echo $i
```

```
done
```

# LINUX – Intro into Scripts

לולאות בתוך הסקריפטים (3) :

```
for (( i=9; i<=12; i++ ))  
do  
    echo $i  
done
```

# LINUX – Intro into Scripts

## LINUX BASH if - elif - else - fi

if [ expression 1 ]

then

Statement(s) to be executed if expression 1 is true

elif [ expression 2 ]

then

Statement(s) to be executed if expression 2 is true

else

Statement(s) to be executed if no expression is true

fi

## Example : if - elif - else - fi

```
#!/bin/sh
a=5
b=100
if [ $a == $b ]
then
    echo "a is equal to b"
elif [ $a -gt $b ]
then
    echo "a is greater than b"
elif [ $a -lt $b ]
then
    echo "a is less than b"
else
    echo "None of the condition met"
fi
```

# LINUX – Intro into Scripts

קליטת נתון מהמשתמש עם בדיקת תקינות הקלט :

```
#!/bin/bash
```

```
#05_CheckAndDisplUserInput_1.sh
```

```
MINVAL=8
```

```
MAXVAL=30
```

```
echo Enter a number that is 1. greater than { $MINVAL } and 2. lesser than { $MAXVAL } :
```

```
read num
```

```
if [ $MINVAL -gt $num ]
```

```
then
```

```
echo "Violation 1. : $num < $MINVAL !!!"
```

```
exit 1
```

```
fi
```



# LINUX – Intro into Scripts

האם הסקריפט מגיב נכון?

```
$ ./05_CheckAndDisplUserInput_1.sh
```

```
Enter a number that is 1. greater than { 8 } and 2. lesser than { 30 } :
```

```
3
```

```
Violation 1. : 3 < 8 !!!
```

```
$ ./05_CheckAndDisplUserInput_1.sh
```

```
Enter a number that is 1. greater than { 8 } and 2. lesser than { 30 } :
```

```
10
```

```
$ ./05_CheckAndDisplUserInput_1.sh
```

```
Enter a number that is 1. greater than { 8 } and 2. lesser than { 30 } :
```

```
34
```

# LINUX – Intro into Scripts

```
$ ./05_CheckAndDisplUserInput_2.sh
```

כך הסקריפט צריך להגיב !!!

```
Enter a number that is 1. greater than { 8 } and 2. lesser than { 30 } :
```

```
3
```

```
Violation 1. : 3 < 8 !!!
```

```
$ ./05_CheckAndDisplUserInput_2.sh
```

```
Enter a number that is 1. greater than { 8 } and 2. lesser than { 30 } :
```

```
10
```

```
The user input 10 is just fine.
```

```
$ ./05_CheckAndDisplUserInput_2.sh
```

```
Enter a number that is 1. greater than { 8 } and 2. lesser than { 30 } :
```

```
34
```

```
Violation 2. : 30 < 34 !!!
```

```
#!/bin/bash
```

```
#05_CheckAndDisplUserInput_2.sh
```

```
MINVAL=8
```

```
MAXVAL=30
```

```
echo "Enter a number that:" { $MINVAL } "<= number <=" { $MAXVAL } ":"
```

```
read num
```

```
if [ $MINVAL -gt $num ]
```

```
then
```

```
echo "Violation 1. : $num < $MINVAL !!!"
```

```
exit 1
```

```
elif [ $MAXVAL -lt $num ]
```

```
then
```

```
echo "Violation 2. : $MAXVAL < $num !!!"
```

```
exit 1
```

```
else
```

```
echo "The user input $num is just fine."
```

```
fi
```

# LINUX – Intro into Scripts

## Arguments with Bash scripts

address\_parameters.sh

-----

```
#!/bin/bash
```

```
echo "Name: $1"
```

```
echo "Street: $2"
```

```
echo "City: $3"
```

```
echo "State" $4"
```

# LINUX – Intro into Scripts

## Arguments with Bash scripts

דוגמה של הרצה :

```
$ ./address_parameters.sh "John Smith" "22335 HiTec St." "Needham" "MS"
```

Name: John Smith

Street: 22335 HiTec St.

City: Needham

State: MS

# LINUX – Intro into Scripts

הסקריפט ShowParamsAmount.sh מציג כמה פרמטרים הוא קיבל:

```
#!/bin/bash
#if (($# == 0))
then
    echo "No parameters were passed"
    exit 1
fi
echo "Number of parameters is " $#
```

\_\_\_\_\_EXAMPLE\_\_\_\_\_

```
$ ./ShowParamsAmount.sh 1 2 three 4
Number of parameters is 4
```

# LINUX – Intro into Scripts

משימה: כתבו סקריפט אשר מקבל שלושה פרמטרים ויוצר שתי שכבות של תיקיות.

זאת התחלת הסקריפט. (המשך המשימה בשקופית הבאה)

```
#!/bin/bash
```

```
#TwoLayersDirs.sh
```

```
#The script builds two layers of directories
```

```
#accordingly to 3 parameters it receives
```

```
dir_name=$1
```

```
first_layer=$2
```

```
second_layer=$3
```

# LINUX – Intro into Scripts

קראו למשתנים : `dir_name` , `first_layer`, `second_layer` .  
אם מספר הפרמטרים שונה משלוש הסקריפט יוציא הודעת שגיאה.  
עוד כללים לאימות את הפרמטרים :

$$0 < \text{first\_layer} < 4$$

$$0 < \text{second\_layer} < 5$$

במידת הצורך יש להציג הודעת שגיאה מתאימה.

בשקופית הבא ראו דוגמאות של ההרצה והתוצאה ...



```
$ ./07_TwoLayersDirs.sh
-E- Actual number of parameters is 0 when requested is 3 :
dir_name | first_layer | second_layer
$ ./07_TwoLayersDirs.sh testDIR 0 5
-E- first_layer = 0. It should be 0 < first_layer < 4.
$ ./07_TwoLayersDirs.sh testDIR 3 5
-E- second_layer = 5. It should be 0 < second_layer < 5.
$ ./07_TwoLayersDirs.sh testDIR 3 4
$ tree
```

```
$ tree
├── testDIR_1
│   ├── testDIR_1_1
│   ├── testDIR_1_2
│   ├── testDIR_1_3
│   └── testDIR_1_4
├── testDIR_2
│   ├── testDIR_2_1
│   ├── testDIR_2_2
│   ├── testDIR_2_3
│   └── testDIR_2_4
└── testDIR_3
    ├── testDIR_3_1
    ├── testDIR_3_2
    ├── testDIR_3_3
    └── testDIR_3_4
```