Contents

1 Basic Test Results

```
Starting tests...
Thu Nov  6 10:17:41 IST 2014
ef773a17542466ed459205dd6863c896ffb92514  -


-rw-r--r-- ransha/stud     566 2014-11-03 21:54 Hello.py
-rw-r--r-- ransha/stud    1910 2014-11-06 10:12 HelloTurtle.py
-rw-r--r-- ransha/stud    4012 2014-11-05 15:32 ex1.txt
-rw-r--r-- ransha/stud    1029 2014-11-05 15:42 README

Testing README...
Done testing README...

Testing Hello.py...
result_code    Hello    correct    1
Done testing Hello.py

Tests completed

Note that the file HelloTurtle.py is not automatically tested
```

פעם הבאה נא
להשתמש בטמפלייט
שבאתר. לא הורדו
נקודות.

2 README

```
ransha
203781000
Ran Shaham

#################################################################
#README for ex1: Hello World and Hello Turtle.                  #
#################################################################
#                                                               #
# Description:                                                   #
# In this exercise I learned how to use the Turtle module a bit  #
# of python programming and learned about the unix enviornment.  #
#                                                               #
#                                                               #
# List of submitted files:                                       #
# 1. Hello.py                                                    #
# 2. HelloTurtle.py                                              #
# 3. ex1.txt                                                     #
# 4. README (this file)                                          #
#################################################################
```

3 Hello.py

```
#####################################################################
#FILE: Hello.py                                                     #
#WRITER: Ran Shaham,ransha,203781000                                #
#EXERCISE: intro2cs ex1 2014-2015                                   #
#DESCRIPTION:                                                      #
#A simple program that prints "Hello World!" to the standard       #
#output (screen).                                                   #
#####################################################################
print("Hello World!")
```

4 HelloTurtle.py

```python
##################################################################
#FILE: HelloTurtle.py                                            #
#WRITER: Ran Shaham,ransha,203781000                             #
#EXERCISE: intro2cs ex1 2014-2015                                #
#DESCRIPTION:                                                    #
#A program that draws some simple geometric shapes on the screen  #
#and prints "Hello Turtle!", using Turtle graphics.              #
##################################################################
import turtle

#title for the display window
turtle.title("Fun with Tutrtle Graphics and Python")
turtle.up()            #lift the pen up, no drawing.
turtle.goto(-100,-100) #move turtle to the absolute position (-100,-100)
turtle.down()          #pen is down, drawing now.

turtle.color("red")    #changes the pen color to red
turtle.goto(100,-100)  #moves the pen only on one axis at each line, thus
                       #creating 4 straight lines in 4 directions (a square)
turtle.goto(100,100)
turtle.goto(-100,100)
turtle.goto(-100,-100) #returned to the initial point.

turtle.up()            #I want to move the turtle without drawing anything so
                       #I lift the pen up.
turtle.goto(0,-100)    #these 4 lines draw an orange circle that it's center
                       # is the absolute point (0,0)
turtle.down()
turtle.color("orange")
turtle.circle(100)

turtle.up()
turtle.goto(-200,0)   #this part creates the bigger blue square.
turtle.down()
turtle.color("blue")
turtle.goto(0,-200)
turtle.goto(200,0)
turtle.goto(0,200)
turtle.goto(-200,0)

#this part writes "Hello Turtle!" in green in the middle of the shapes the
# turtle drew (the command was given in the ex. instructions...)
turtle.up()
turtle.goto(-70,-5)
turtle.down()
turtle.color("green")
turtle.write("Hello Turtle!", font=("Arial", 20, "normal"))
#done
turtle.done()
```

קצת יותר מידי תיעוד, מספיק בתחילת כל קטע להסביר מה הקטע עושה, לדוגמא שורה מספר 41

5 ex1.txt

```
PART 3

1. help(turtle.goto) : A detailed help screen that explains the goto function in the turtle module,
describing the possible arguments for this function and uses.

2. turtle.goto : The shell prints that my line was a funcion, followed by "at" and a long
hexadecimal number (I do not know what the number after "at" means).
"<function goto at 0x7f0f6c2e6840>"

3. turtle.goto(100,100) : A graphics window has opened and a black line was drawn from
the center of it to a righter-upper point.

PART 4 - UNIX ENVIRONMENT

1. mkdir : creates an empty directory with a given name.
rmdir : deletes an empty directory with a given name.
cd : changes the directory to a chosen path.
cd ~ : changes the directory to the home directory of the current user (it can be used to changed
directory to other user's home dir- for example: cd ~other_user).
cd ~/: does the same as cd ~, because '~' is the user's home dir and if the / isn't followed by a path
it's the same as writing the command without it.
(if you meant cd / : it changes the directory to the root directory in the file system.)

2. The directory '.' is the current directory. the '..' one is the parent directory
(one step up in the file system). It can be useful when you want to copy files
from a parent directory to the current one. "cp ../file.tar ." - copies a file named
'file.tar' from parent to current directory.

3. relative path is a path that depends on your current location.
(if i'm at home dir: "safe/intro2cs..." is a relative path) and an absolute path is a
full path that describes the exact location in the filesystem regardless of the current path ("/cs/stud/ransha/safe/intro2cs

4. The signs '*' and '?' are used if we don't want to enter specific characters.
* is used when it's a number of characters that we don't specify (can also be one), and the ?
replaces a specific one. for example "ls *.??" prints a list of files that consists of
any number of characters as the file's name (before the '.') and only two characters for type
(after the '.'). in the case of ex1 directory (as it is now), it prints the list of Hello.py and HelloTurtle.py.
it can also be used for the cp command. for example 'cp *.py ~/Documents' (copies all .py files
to the Documents directory).

5. the & sign is made to run a program in the background, thus allowing us to keep
using the shell. If we forgot this sign we can press CTRL + C to terminate the
running app in the foreground, or alternatively press CTRL + Z to  suspend it and then
type 'bg' to send it to the background and continue using the shell.

6. the permissions for a file is written after it's type (- for file/d for directory) when you run
the ls -l command (the file name is in the end of the line). it is described
by 3 triplets of rwx, the first one describes the owner's permissions (read,write,execute),
the second to describe group's and the last to describe others' permissions.
for example: -rw-rw---- is a file that it's owner and it's group can read and write.

7. grep -n 'turtle.goto' safe/intro2cs/ex1/HelloTurtle.py :
when this command is run from home dir, it prints to the console all the lines that contain the phrase
'turtle.goto' inside the file 'HelloTurtle.py' and the number of the line the pattern appears in (-n).

cat ex1.txt :
when inside the ex1 dir, it prints the contents of the text file 'ex1.txt' to the console.
(also works with other simple files such as *.py)
```

זה מיקום הפונקציה
בזיכרון -1

```
diff -y ex0 ex1 :
when in intro2cs dir, it prints which files are uniqe to each dir ('Only in ex0: ex0.tar')
and the difference between files that has the same names in these directories (Hello.py, HelloTurtle.py)
(it shows which lines are missing in each file) - all of these - in 2 columns (-y).

wc -l HelloTurtle.py
prints to the console the number of lines (newlines) in the file specified, followed
by the file name ('50 HelloTurtle.py'). without the -l, it will also output the bytes and
words count.

cal - prints a simple monthly calander that highlights today.
```