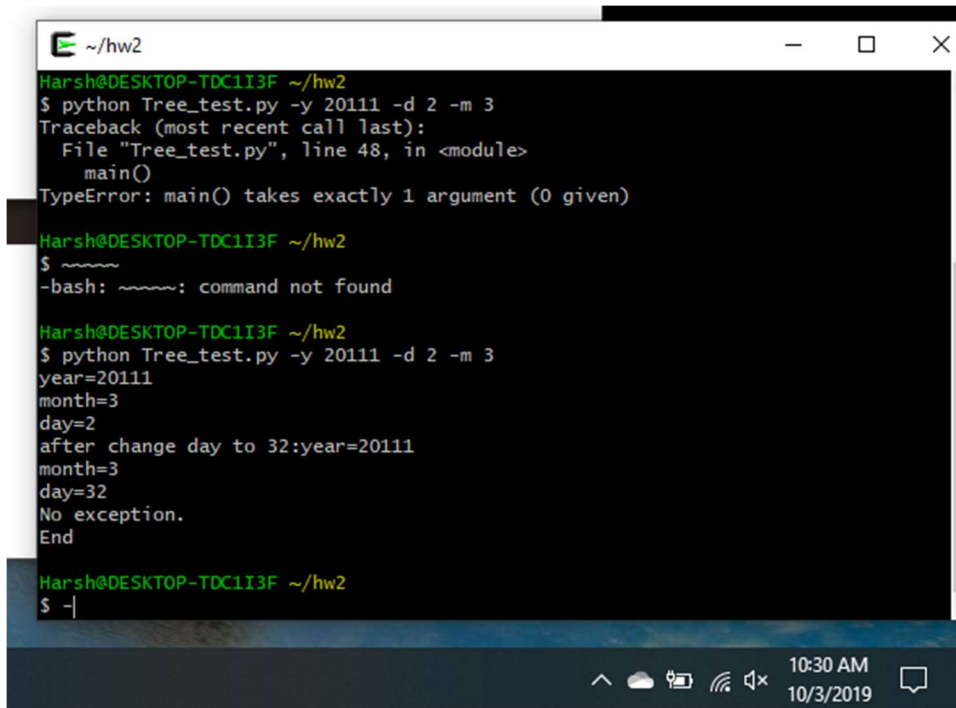


## Week 2:-

### Execution Result:-



```
Harsh@DESKTOP-TDC1I3F ~/hw2
$ python Tree_test.py -y 20111 -d 2 -m 3
Traceback (most recent call last):
  File "Tree_test.py", line 48, in <module>
    main()
TypeError: main() takes exactly 1 argument (0 given)

Harsh@DESKTOP-TDC1I3F ~/hw2
$ ~~~~~
-bash: ~~~~~: command not found

Harsh@DESKTOP-TDC1I3F ~/hw2
$ python Tree_test.py -y 20111 -d 2 -m 3
year=20111
month=3
day=2
after change day to 32:year=20111
month=3
day=32
No exception.
End

Harsh@DESKTOP-TDC1I3F ~/hw2
$ -|
```

### Tree.py:-

```
class ErrorCode(Exception):
```

```
    """The base class of erros"""
```

```
    #__code=0
```

```
# constructor
```

```
def __init__(self, n):
```

```
    self.__code = n
```

```
def getCode(self):
```

```
    return self.__code
```

```
class ErrorYear(ErrorCode):  
    """Raised when the input value is Negative"""  
    #__num=0  
  
    # constructor  
    def __init__(self, n):  
        ErrorCode.__init__(self, 1)  
        self.__num = n  
    def getNum(self):  
        return self.__num
```

```
class ErrorDay(ErrorCode):  
    """Raised when the input value is more than 130"""  
    #__num=0  
  
    # constructor  
    def __init__(self, n):  
        ErrorCode.__init__(self, 2)  
        self.__num = n  
    def getNum(self):  
        return self.__num
```

```

class ErrorMonth(ErrorCode):
    """Raised when the input value is more than 130"""
    #__num=0

    # constructor
    def __init__(self, n):
        ErrorCode.__init__(self, 3)
        self.__num = n
    def getNum(self):
        return self.__num

```

```

class Tree:

    #####
    # Helping function
    #####

    def __trace(self, s):
        print(s)

    #####
    # Manager function
    #####
    # Including a default contrutctor

```

```
def __init__(self, y,d,m):
    if y < 0:
        raise ErrorYear(y)
    elif (d < 0) + (d > 31):
        raise ErrorDay(d)
    elif (m < 0) + (m > 12):
        raise ErrorMonth(m)
    self.__year = y
    self.__day = d
    self.__month = m
def __del__(self):
    pass
```

```
#####
```

```
# Access function
```

```
#####
```

```
def getYear(self):
```

```
    return self.__year
```

```
def setYear(self, y):
```

```
    self.__year = y
```

```
def getDay(self):
```

```
    return self.__day
```

```

def setDay(self, d):
    self.__day = d
def getMonth(self):
    return self.__month
def setMonth(self, m):
    self.__month = m
def isCentennial(self):
    return self.__age %100 == 0

#####
# Implementor function
#####
def toString(self):
    return ("year=" + str(self.__year)+ "\n" \
    + "month=" + str(self.__month)+ "\n" \
    + "day=" + str(self.__day))
def reset(self):
    year = 0
    month = 1
    day = 1

```

**Tree\_test.py:-**

```
#!/usr/bin/python
```

```
# Tree_test.py

import Tree

import sys, getopt

def usage():

    print ('Usage: Triangle.py -h')

    print ('Usage: Triangle.py -y -d -m ')

    print ('Usage: Triangle.py --year=<year> --day=<day> --month=<month>')

def main(argv):

    year = ""

    day = ""

    month = ""

    try:

        opts, args = getopt.getopt(argv,"hy:d:m:",["year=", "day=", "month="])

    except getopt.GetoptError:

        usage()

        sys.exit(2)

    for opt, arg in opts:

        if opt == '-h':
```

```
        usage()
        sys.exit()
    elif opt in ("-y", "--year"):
        year = arg
    elif opt in ("-d", "--day"):
        day = arg
    elif opt in ("-m", "--month"):
        month = arg

    try:
        t1 = Tree.Tree(int(year),int(day),int(month))
        print(t1.toString())
        t1.setDay(32)
        print("after change day to 32:" + t1.toString())

    except Tree.ErrorYear as obj:
        print("Error", obj.getCode(),": The year", obj.getNum(), "is not
right.")
    except Tree.ErrorDay as obj:
        print("Error", obj.getCode(),": The day", obj.getNum(), "is not
right.")
    except Tree.ErrorMonth as obj:
```

```
    print("Error", obj.getCode(), ": The month", obj.getNum(), "is not  
right.")
```

```
else:
```

```
    print("No exception.")
```

```
finally:
```

```
    print("End")
```

```
if __name__ == '__main__':
```

```
    main(sys.argv[1:])
```