CS2004

Assignment 2

Anushervon Rakhmatov

201356375

Table of Contents

Description of the programming problem:

Non-Deterministic Behavior of a Concurrent System

1. Write a routine which given a Seed and a Range limit, both integers, will generate a random sequence of integers , such that the value of each element in the sequence is within 1 to Range.

As discussed in class, you can impose more ”randomness” in the sequence by making the Seed itself generated at random [This can be done in several possible ways: eg, using a system clock as the Seed, or soliciting, interactively the value of the Seed during each run.

2. For this assignment , write three simple routines, printA, printB, and printC, where printA, printB, and printC print character “A”, “B” and ”C” respectively.

3. Write a Scheduler, which will use the items generated by the random number generator, and depending on the current value of the item, will invoke either the printA or the printB, or printC.

User Instructions

User-friendly description.

The program is a simulator of the concurrent processing system, where multiple processes run independently of each other, but not simultaneously.

System requirements:

OS: Windows/Linux

IDE: Python IDE

Programming Language: Python

Console requirements: Make sure that Python 3.6 or later is installed.

How to run

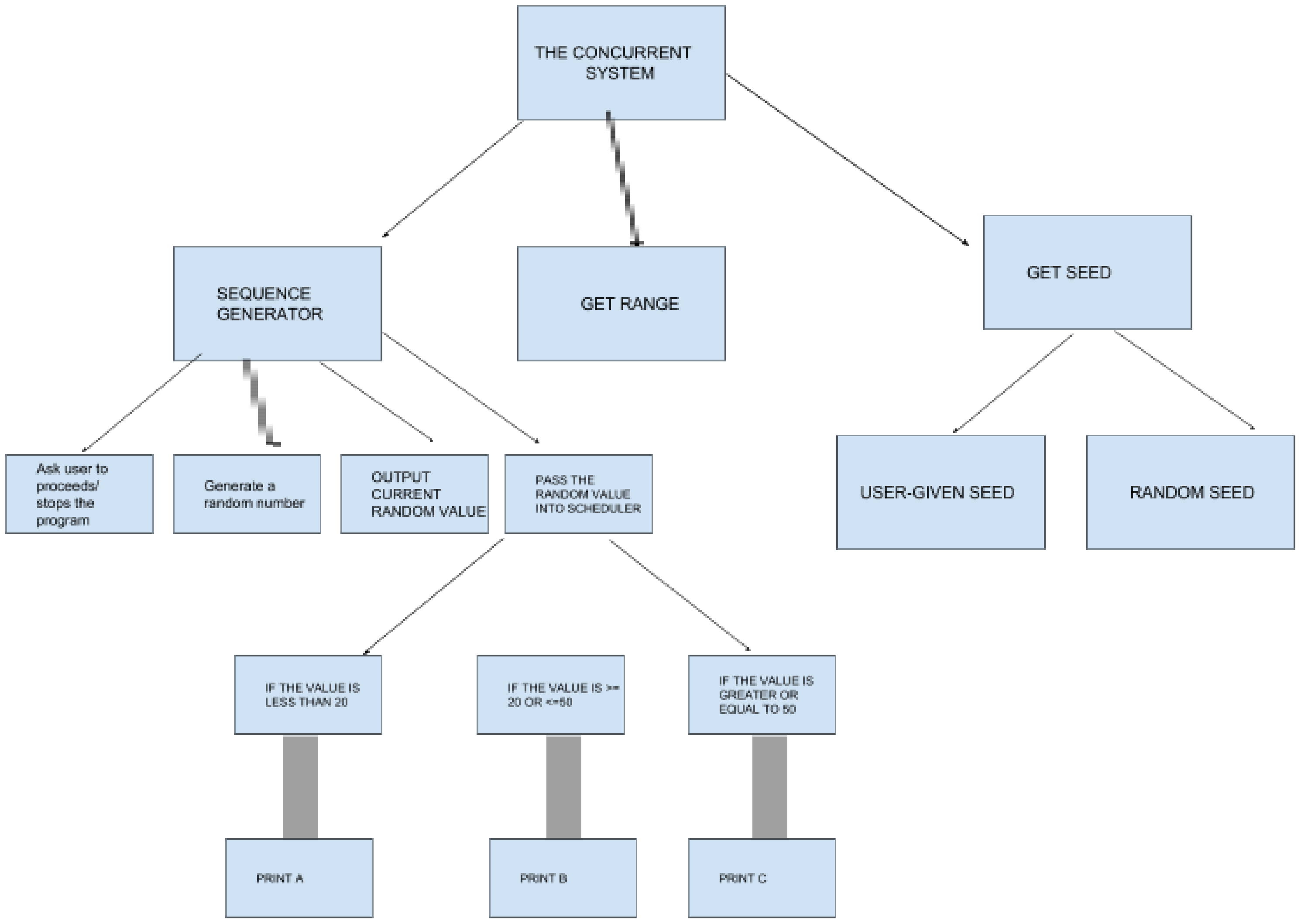
For Linux:

1. Insert the USB into your computer and locate the directory Assignment 2.
2. Open the Terminal and navigate to the Assignment 2 directory.
3. Type*:* *python3 scheduler.py* to start the program.
4. Refer to the **Program execution** section of this document for further steps.

For Windows:

1. Insert the USB into your computer and locate the directory Assignment 2.
2. Right click *the scheduler.py* and select *>Open with>Python IDE3*
3. Refer to the **Program execution** section for further steps.

Top Down Design

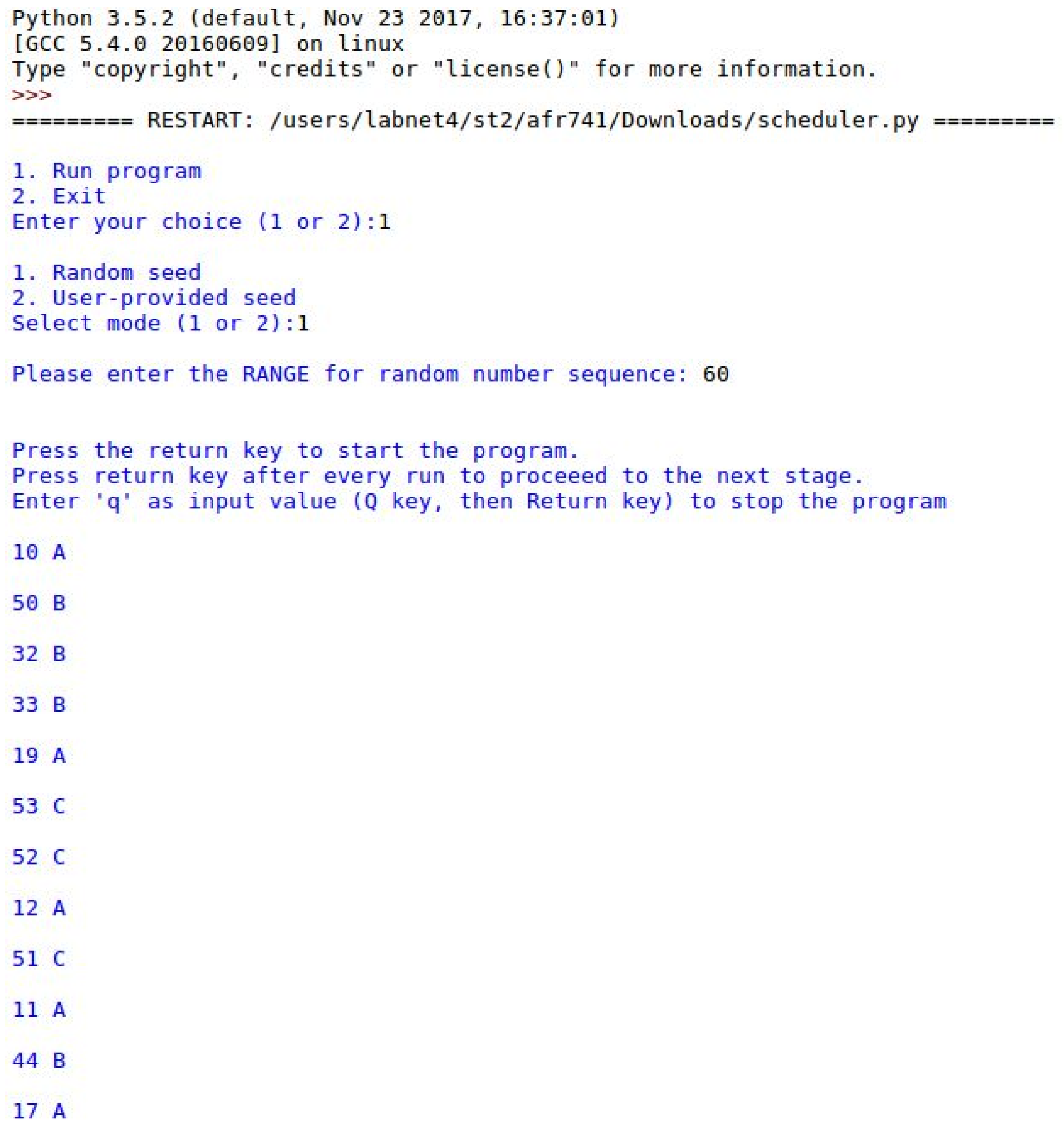

THE CONCURRENT SYSTEM
SEQUENCE GENERATOR
GET RANGE
GET SEED
Ask user to proceeds/ stops the program
Generate a random number
OUTPUT CURRENT RANDOM VALUE
PASS THE RANDOM VALUE INTO SCHEDULER
USER-GIVEN SEED
RANDOM SEED
IF THE VALUE IS LESS THAN 20
IF THE VALUE IS >= 20 OR <=50
IF THE VALUE IS GREATER OR EQUAL TO 50
PRINT A
PRINT B
PRINT C

The Code

```
from random import randint, seed


#Calls function printA() if value is even, otherwise calles function printB()
def scheduler (value):
    if value <= 20:
        printA()
    elif value > 20 and value <=50:
        printB()
    elif value >50:
        printC()




#Prints "A" to the screen on every call
def printA():
    print ("A")


#Prints "B" to the screen on every call
def printB():
    print ("B")

#Prints "C" to the screen on every call
def printC():
    print ("C")


#Generates and prints a random value between 1 and limit, based on valueOfSeed. Makes call to scheduler with the ra
def randNumSeq(valueOfSeed, limit):
    seed(valueOfSeed) #Sets the seed value for randint()
    print("\n\nPress the return key to start the program.\nPress return key after every run to proceeed to the next
    while input().lower() != 'q':
        value = randint(1, limit)
        print (value, end=" ")
        scheduler(value)




def main ():
    #lets the user to run multiple instances of the program with various settings
    while True:
        option = input("\n1. Run program\n2. Exit\nEnter your choice (1 or 2):")
        while option != "1" and option != "2":
            option = input("\nWrong input. Enter either 1 or 2:")
        if option == "2": #Exits the program based on user's choice
            return

    #set the value of seed and allows the user to select between random seed mode and User-provided seed mode
        option = input("\n1. Random seed\n2. User-provided seed\nSelect mode (1 or 2):")
        while option != "1" and option != "2" and option != "3":
            option = input("\nWrong input. Enter either 1 or 2 or 3:")
        if option == "2":
            valueOfSeed = int(input("\nEnter any positive integer for the seed value of the random number genera
        else:
            valueOfSeed = (randint(1,200))
 #Set range value
        limit = int(input("\nPlease enter the RANGE for random number sequence: "))
        randNumSeq(valueOfSeed, limit)
main()
```

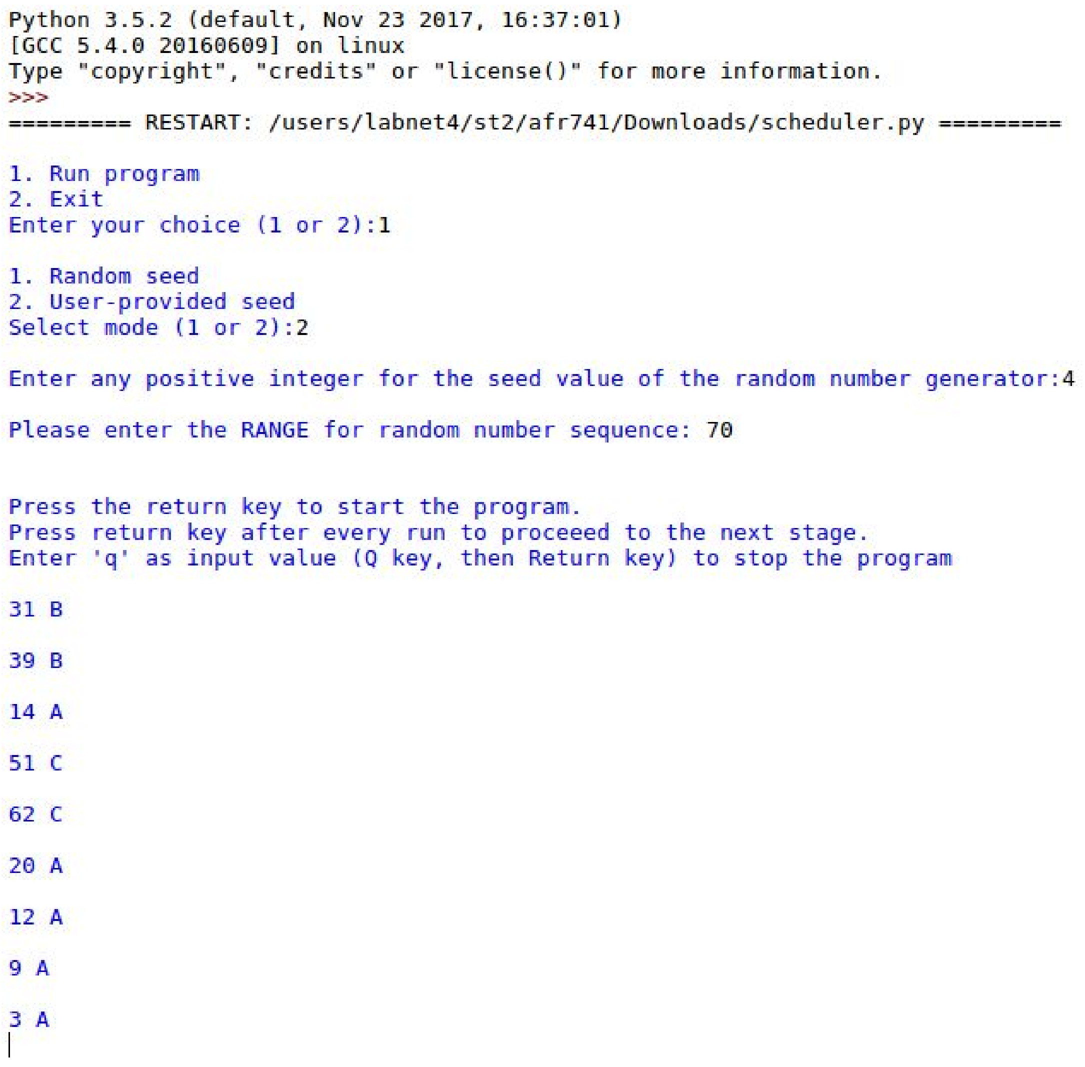
Screenshots of the program execution

- Executing “Random Seed” option



```
Python 3.5.2 (default, Nov 23 2017, 16:37:01)
[GCC 5.4.0 20160609] on linux
Type "copyright", "credits" or "license()" for more information.
>>>
========= RESTART: /users/labnet4/st2/afr741/Downloads/scheduler.py =========

1. Run program
2. Exit
Enter your choice (1 or 2):1

1. Random seed
2. User-provided seed
Select mode (1 or 2):1

Please enter the RANGE for random number sequence: 60


Press the return key to start the program.
Press return key after every run to proceeed to the next stage.
Enter 'q' as input value (Q key, then Return key) to stop the program

10 A

50 B

32 B

33 B

19 A

53 C

52 C

12 A

51 C

11 A

44 B

17 A
```

- Executing “User provided seed” option



```
Python 3.5.2 (default, Nov 23 2017, 16:37:01)
[GCC 5.4.0 20160609] on linux
Type "copyright", "credits" or "license()" for more information.
>>>
========= RESTART: /users/labnet4/st2/afr741/Downloads/scheduler.py =========

1. Run program
2. Exit
Enter your choice (1 or 2):1

1. Random seed
2. User-provided seed
Select mode (1 or 2):2

Enter any positive integer for the seed value of the random number generator:4

Please enter the RANGE for random number sequence: 70


Press the return key to start the program.
Press return key after every run to proceeed to the next stage.
Enter 'q' as input value (Q key, then Return key) to stop the program

31 B

39 B

14 A

51 C

62 C

20 A

12 A

9 A

3 A
```

Dictionary of symbolic names

Variable name	Type	function usage	Description
scheduler()	function	scheduler()	This function acts as a simulator of the scheduler in operating systems. Here it “schedules” (calls) our two processes printA, printB and printC based on whether value is less than 20, great than 20 and less than 50, or greater than 50.
printA()	function	scheduler(),printA()	The function prints the character A
printB()	function	scheduler(), printB()	The function prints character B
printC()	function	scheduler(), printC()	The function prints character C
randNumSeq()	function	randNumSeq()	This function sets the seed using the function seed from the random module. It then starts a loop that is terminated only when the user enters ‘Q’ (caseinsensitive). Otherwise the user may just hit the Return key to continue the loop. On each iteration of the loop, randint(1, limit) generates a

			pseudo-random number between 1 and limit. It then prints the random number and passes the number to the scheduler.
valueOfSeed	integer	main(), randNumSeq()	Takes the value of seed given by a user
limit	integer	main(), randNumSeq()	Takes a range of the sequence given by a user
option	integer	main()	Hold the option number selected by a user in the menu

Limitations and Improvements

This program is a very simple demo of the concurrent systems switching between the processes. n practice, there is more to it than just generating a random number and deciding which process to run. It is however, a good introduction to concurrent processing.

Another limitation is that the random number generation is pseudo. Which means that there is an algorithm behind it, therefore, it is not completely random. An alternative approach is to use the system clock to generate the number itself.