German University in Cairo
Faculty of Media Engineering and Technology
Prof. Dr. Slim Abdennadher
Dr. Aysha ElSafty

Introduction to Computer Science, Winter Semester 2016
Practice Assignment 4

Discussion: 29.10.2016 - 3.11.2016

Exercise 4-1
To Be Discussed

Given the following iterative algorithm:

```
n = eval(input())

previous = -1
result = 1
i = 0

while i <= n:
␣ sum = result + previous
␣ previous = result
␣ result = sum
␣ i = i + 1

print("The fibonacci value is ")
print(result)
```

Trace the algorithm for `n = 6`

Solution:

n	previous	result	i	sum
6	-1	1	0	
6	1	0	1	0
6	0	1	2	1
6	1	1	3	1
6	1	2	4	2
6	2	3	5	3
6	3	5	6	5
6	5	8	7	8

Exercise 4-2 Power By Multiply
To Be Discussed

Write an algorithm that gets two positive integer numbers X and Y and calculates X^Y. Use only the addition and the multiplication.

Solution:

```
x = eval(input())
y = eval(input())

counter = 1
power = 1
```

```
while counter <= y:
␣ power = (power * x)
␣ counter = counter+1

print(power)
```

Exercise 4-3

To Be Discussed

- Trace the following iterative algorithm for `num1 = 987, num1 = 64, num1 = -1`:

```
num1 = eval(input())
count = 0
num2 = num1

while ( num1 != -1):
␣ if(num1 == 0):
␣␣ count = 1
␣ if(num2 != 0):
␣␣ count += 1
␣␣ num2 = num2 // 10
␣ else:
␣␣ print(count)
␣␣ count = 0
␣␣ num1 = eval(input())
␣␣ num2 = num1

print("you entered -1")
```

Solution:

num1	count	num2
987	0	987
987	1	98
987	2	9
987	3	0
64	0	64
64	1	6
64	2	0
-1	0	-1

- What is the functionality of this algorithm? i.e What does this algorithm do.

Solution:

This algorithm gets inputs from the user and displays the number of digits of each input number, until the user enters -1.

Exercise 4-4

Write an algorithm that given `n` calculates and displays the value of the following series

$$1^2 + 2^2 + 3^2 + \ldots + n^2$$

Solution:

```
n = eval(input())

sum = 0
counter = 1
while (counter <= n):
␣ sum = sum + (counter * counter)
␣ counter = counter + 1

print(sum)
```

Exercise 4-5

Write an algorithm to find the least number of terms that must be added to the following series

$$1^2 + 2^2 + 3^2 + 4^2 + \dots$$

to give a sum ≥ 10000

Solution:

```
sum = 0
counter = 1
while (sum < 10000):
␣ sum = (sum + counter * counter)
␣ counter = counter + 1

print(" to give a sum >= 10000 we need following number of iterations: ")
print(counter - 1)
```

Exercise 4-6 The product of N

Write an algorithm to calculate the product of n terms in the following series

$$1 * \frac{3}{2^2} * \frac{5}{3^2} * \frac{7}{4^2} * ..... * \frac{X}{n^2}$$

where n is an input value. The algorithm should print the result of the product and the value of X as well. For example: for n=3, the product is $1 * \frac{3}{2^2} * \frac{5}{3^2}$ and the value of X will be 5.

Solution:

```
n = eval(input())
i = 1
prod = 1
while (i<=n):
␣ prod = prod*((2*i)-1) / (i*i)
␣ i = i + 1
print(prod)
print(2*n-1)
```

Another Solution:

```
n = eval(input())
```

```
i = 1
prod = 1
j = 1
while (i<= n):
␣ prod = prod*j / (i*i)
␣ j = j + 2
␣ i = i + 1
print(prod)
print(j-2)
```

Exercise 4-7 Get The GCD
To Be Discussed

The Euclidean algorithm determines the greatest common divisor (GCD) of two positive numbers by repeatedly replacing the larger number with the result of subtracting the smaller one from it until the two numbers are equal.

Write an algorithm for the Euclidean algorithm.

Solution:

```
Num1 = eval(input())
Num2 = eval(input())

while (Num1 != Num2):
␣ if (Num1 > Num2):
␣␣ Num1 = (Num1-Num2)
␣ else:
␣␣ Num2 = (Num2-Num1)

print(Num1)
```

Exercise 4-8 Sum of Even Digits
To Be Discussed

Write an algorithm that sums up the even digits in a given number.

Solution:

```
num = eval(input())
sum = 0
while(num > 0):
␣ if(num%2 == 0):
␣␣ sum += num%10
␣ num = num//10

print(sum)
```

Exercise 4-9 How Many Year Must You Wait

A bank is offering an annual compound interest of 5%. Write an algorithm that tells you how many years you need to wait in order to reach a ballance of y dollars, given your current ballance.

As an Example
if your ballance is 10 000, and your taget is 20 000 you will wait 15 years

Solution:

```
balance = eval(input())
target = eval(input())

years = 0
while balance < target:
␣ balance = balance + balance*0.05
␣ years = years + 1

print(years)
```

Exercise 4-10 Guess The Secret Number

This guessing game is a game where the program selects a random number between 1 and 10 and keeps asking you to guess, if you get it wrong you get a message informing you that you are a bad guesser, and then it asks you to guess again, if you guess right your program congratulates your player and exits the game loop.

As an example assuming the program picked 6 as the secret number:

The program prints **Guess the right number:**
You enter **3**
The program prints **You guessed wrong XD**
You enter **6**
The program prints **You did it the correct number is 6**

Solution:

```
import random

secret = random.randint(0,10)
print("Guess the right number:")
n = eval(input())

while (secret != n):
␣ print("You guessed wrong")
␣ n = eval(input())

print("You did it the correct number is ",n)
```

Exercise 4-11 Prime Number

Write an algorithm that given an integer number determines whether the number is a prime number. A prime number is a number that can only be divided by itself and 1. For example, 11 is a prime number because it can be only divided by 1 and 11. However, 8 is not a prime number since it can be divided by 1, 2, 4 and 8.

Solution:

```
number = eval(input())
```

```
result = True

if (number == 1):
␣ result = True
else:
␣ if (number == 0):
␣␣ result = False
i = 2

while (i < number):
␣ if(number % i == 0):
␣␣ result = False
␣ i = i + 1

print(result)
```

Exercise 4-12 Perfect Number

A perfect integer is a positive integer that is equal to the sum of all of its proper divisors. A proper divisor of an integer n is an integer between 1 (inclusive) and n (exclusive) that divides n with no remainder. For example, 6 is a perfect number because $6 = 1 + 2 + 3$.

Write an algorithm that decides whether an integer is a perfect number.

Solution:

```
n = eval(input())
counter = 2
sumOfFactors = 1
while(counter < n):
␣ if(n % counter == 0):
␣␣ sumOfFactors = sumOfFactors + counter
␣ counter = counter + 1

if(sumOfFactors == n):
␣ print("The number is perfect")
else:
␣ print("The number is not perfect")
```

Exercise 4-13 Reverse Number

Write a function that given an integer number, returns the reverse of this number.

Solution:

-
  ```
  def numDigits(x):
  ␣ if (x == 0):
  ␣␣ c = 1
  ␣ else:
  ␣␣ c = 0
  ␣␣ while (x != 0):
  ␣␣␣ c += 1
  ␣␣␣ x //= 10
  ␣ return c

  def reverse(x):
  ␣ d = numDigits(x) - 1
  ␣ s = 0
  ```

```
␣ while (d >= 0):
␣␣ s += x%10 * (10**d)
␣␣ x //= 10
␣␣ d -= 1
␣ return s

x = eval(input())
print(reverse(x))
```

-
  ```
  x = eval(input())
  r = 0
  while(x != 0):
  ␣ r = r*10 + x%10
  ␣ x //= 10
  print(r)
  ```

Exercise 4-14 Mirror

Write an algorithm that given two integer numbers, checks whether one number is a mirror of the other number. **Hint: Use the reverse function that you implemented in the previous question.**

Solution:

```
def checkMirror(x,y):
␣ return (x == reverse(y))

x = eval(input())
y = eval(input())

print(checkMirror(x,y))
```