Computer Programming
Bachelor in Biomedical Engineering
Bachelor in Applied Mathematics and Computing

Course 2020 / 2021

Exercise Sheet 7

Functions

-SOLUTIONS-

Content Table

Exercise 1 2

Exercise 2 2

Exercise 3 3

Exercise 4 4

Exercise 5 5

Exercise 6 7

Exercise 1

a) Write a function ‘obtainSeconds’ that receives as parameters three numbers representing hours, minutes and seconds; and returns the total number of seconds.

b) Write a program that asks the user to introduce hours, minutes and seconds, calls the function ‘obtainSeconds’, and prints the number of total seconds on screen.

Example of execution:

```
Introduce number of hours: 1
Introduce number of minutes: 1
Introduce number of seconds: 1
The total number of seconds is 3661
```

SOLUTION TO A/FUNCTION

```
function [totalseconds] = obtainSeconds(hours, minutes, seconds) %
function obtainSeconds
totalseconds = hours*3600 + minutes*60 + seconds;
end % always at the end of function
```

SOLUTION TO B/MAIN PROGRAM

```
clear;
hours = input('Introduce number of hours: ');
minutes = input('Introduce number of minutes: ');
seconds = input('Introduce number of seconds: ');
total = obtainSeconds(hours, minutes, seconds);
fprintf('The total number of seconds is %d\n', total);
```

Exercise 2

Write a function called ‘polynom’ that implements the following formula:

$$y(x) = a*x^{2}+b*x+ c$$

The function receives as parameters the values of a, b, c and x, and returns the corresponding value y(x).

Then, write a program (script) that calls the function and generates the following output:

```
Introduce a: 2
Introduce b: 1
Introduce c: 5
```

```
Your function is y(x) = 2x2 + 1x + 5
Evaluate for x: 10
y(10) = 215
```

FUNCTION

```
function [y] = polynom(a, b, c, x)
% function polynom computes the value of y following the formula y =
a*x^2+b*x+c;
y = a*x^2+b*x+c;
end
```

MAIN PROGRAM

```
clear;
va = input('Introduce a: ');
vb = input('Introduce b: ');
vc = input('Introduce c: ');
fprintf('Your function is y(x) = %dx2 + %dx + %d\n', va, vb, vc);
vx = input('Evaluate for x: ');
vy = polynom(va, vb, vc, vx);
fprintf('y(%d) = %d\n', vx, vy);
```

Exercise 3

Write a function ‘obtainTime’ that receives a number representing a total of seconds and returns the corresponding hours, minutes and seconds.

Test the function writing a program that asks the user to introduce a number of seconds and prints the corresponding hours, minutes and seconds on screen.

Example of execution:

```
Introduce the total number of seconds: 3661
Hours: 1
Minutes: 1
Seconds: 1
```

FUNCTION

```
function [hours, minutes, seconds] = obtainTime(totalSeconds)
% function obtainTime
hours = floor(totalSeconds / 3600);
restSeconds = rem(totalSeconds, 3600);
minutes = floor(restSeconds / 60);
seconds = rem(restSeconds, 60);
end
```

MAIN PROGRAM

```
clear;
vtotalSeconds = input('Introduce the total number of seconds: ');
[varHours, varMinutes, varSeconds] = obtainTime(vtotalSeconds);
fprintf('Hours: %d\nMinutes: %d\nSeconds %d\n', varHours, varMinutes,
varSeconds);
```

Exercise 4

Write a function that, given a 9-element vector, returns a 3x3 matrix. The first 3 elements of the vector correspond to the first row of the matrix, the next 3 elements correspond to the second row and the last 3 correspond to third row.

Then write a program that uses the function. The program asks the user to introduce the elements of a 9-element vector, calls the function, and prints screen the resulting matrix on screen.

```
Introduce component (1): 1
Introduce component (2): 2
Introduce component (3): 3
Introduce component (4): 4
Introduce component (5): 5
Introduce component (6): 6
Introduce component (7): 7
Introduce component (8): 8
Introduce component (9): 9
The 3x3 matrix is:
row1: 1 2 3
row2: 4 5 6
row3: 7 8 9
```

FUNCTION

```
function [A] = fill3x3Matrix(vector)
A = zeros(3,3);
index = 0;
for row = 1:3
    for column = 1:3
        index = index + 1;
        A(row,column) = vector(index);
    end
end
end
```

MAIN PROGRAM

```
clear;
rows = 3;
columns = 3;
vectorLength = rows*columns;
v = zeros(1, vectorLength);
for i = 1:vectorLength
    fprintf('Introduce component (%d): ',i);
    v(i) = input('');
end

M = fill3x3Matrix(v);

for i = 1:3
    fprintf('row%d: ', i);
    for j=1:3
        fprintf('%d ', M(i,j));
    end
    fprintf('\n');
end
```

Exercise 5

a) Write a function that receives a number and returns a vector containing its factors

b) Write a function that receives a number and returns 1 if the number is a prime and 0 otherwise

c) Write a function that receives a number and returns a vector containing its prime factors

d) Write a program that asks the user to introduce numbers and prints its prime factors on screen. The program stops when the user introduces a negative number.

Example of execution:

```
Introduce a number: 16
Prime factors: 1 2
Introduce a number: 100
Prime factors: 1 2 5
Introduce a number: -1
Bye!
```

FUNCTIONS

```
% Get all factors of a value
function [vfactors] = factors (number)
vfactors = [];
for value = 1:number
    if (rem(number, value) == 0)
        vfactors = [vfactors value];
    end
end
end
```

```
% Determine whether a number is a prime
function [result] = isPrime (number)
i = 2;
while ((i < (number/2)) && (rem(number,i) ~= 0))
    i = i + 1;
end
if (i < (number/2) || (number == 4))
    result = 0;
else
    result = 1;
end
end
```

```
% Get prime factors of a value
function [vprfacts] = primeFactors (number)
vprfacts = [];
cont = 0;
vector = factors(number); % call function factors to calculate all
factors
for val= vector
    if isPrime(val) ==  1 % call function isPrime to determine which of
the factors is a prime
        cont = cont + 1;
        vprfacts(cont) = val;
    end
end
end
```

MAIN PROGRAM

```
clear;
value = input('Introduce a number: ');
while (value >= 0)
    vfacts = primeFactors(value);
    fprintf('Prime factors: ');
    for i = vfacts
        fprintf('%d ', i);
    end
    fprintf('\n');
    value = input('Introduce a number: ');
end
disp('Bye!');
```

Exercise 6

Write a function called ‘generateRandom’ that generates a randomly generated natural number given two input natural numbers (minimum value and maximum value). The generated number will be in the range [minimum value, maximum value] (both included in the range).
You can use the MATLAB functions **rand** and **floor** to solve this exercise.

Note: The rand function returns a random number between 0 and 1. You have to proportionally transform this result to the interval between the two values that your function receives as parameters.

Test your function by filling a vector with 20 randomly generated values in the range [2,4] and then print the content of the vector.

Example:

```
34342424422244344233
```

FUNCTION

```
function [result] = generateRandom(minVal,maxVal)
range = (maxVal-minVal) + 1;
result = minVal + floor(rand()*range);
end
```

MAIN PROGRAM

```
clear;
vlength = 20;
vector = zeros(1,vlength);
for i = 1:vlength
    vector(i) = generateRandom(2,4);
end
for i = 1:vlength
    fprintf('%d ', vector(i));
end
fprintf('\n');
```