Extra exercise 1

Write a program that simulates throwing a coin until you get the same side (heads or tails) three times in a row.

TIP: The function rand() will generate a random value from 0 to 1. Use *help rand* to get more information about this function. Keep in mind that the probability of getting each side of the coin should be the same.

Example of execution:

```
Throw: 1 Result: HEADS
Throw: 2 Result: TAILS
Throw: 3 Result: HEADS
Throw: 4 Result: TAILS
Throw: 5 Result: HEADS
Throw: 6 Result: HEADS
Throw: 7 Result: TAILS
Throw: 8 Result: TAILS
Throw: 9 Result: HEADS
Throw: 10Result: TAILS
Throw: 11Result: TAILS
Throw: 12Result: TAILS
```

SOLUTION

```
clear;
last = 0; % 0 represents heads, 1 represents tails
times = 0;
throw = 1;
while (times < 3)
    fprintf('Throw: %d \t Result: ', throw); % number of 'throw'
    random = rand();
    if (random < 0.5)
        coin = 0; % heads
        disp('HEADS');
    else
        coin = 1; % tails
        disp('TAILS');
    end
    if coin == last % check if for this throw we got the same side as
for the throw before
        times = times +1;
    else
        times = 1;
        last = coin;
    end
    throw = throw + 1;
end
```

Extra exercise 2

Write a program that prints the numbers introduced by the user until a 0 is given. In the end, it prints the three biggest numbers.

Example of execution:

```
Introduce a number: 10
The number is 10
Introduce a number: 24
The number is 24
Introduce a number: 9
The number is 9
Introduce a number: 20
The number is 20
Introduce a number: 0
The three biggest numbers are 24, 20 and 10
```

SOLUTION

```
clear;
maxNum1 = 0; % the biggest number
maxNum2 = 0; % the second biggest number
maxNum3 = 0; % the third biggest number
numberRead = input('Introduce a number: '); % intiliziation of
numberRead
while (numberRead ~= 0)
    fprintf('The number is %d\n', numberRead);
    if (numberRead ~= maxNum1) && (numberRead ~= maxNum2) &&
(numberRead ~= maxNum3)
        if (numberRead > maxNum1)
            maxNum3= maxNum2;
            maxNum2= maxNum1;
            maxNum1= numberRead;
        elseif (numberRead > maxNum2)
            maxNum3= maxNum2;
            maxNum2= numberRead;
        elseif (numberRead > maxNum1)
            maxNum3= numberRead;
        end
    end
    numberRead = input('Introduce a number: ');
end
fprintf('The three biggest numbers are %d %d %d\n', maxNum1, maxNum2,
maxNum3);
```

Extra exercise 3

Write a program that asks the user to introduce a series of numbers, one after the other, until the user introduces a 0. Then, the program tells the user if the list of numbers introduced is the same as producing these numbers in reverse order.

Examples of execution:

```
Introduce a number: 1
Introduce a number: 2
Introduce a number: 2
Introduce a number: 1
Introduce a number: 0

The list of numbers is the same as the numbers in reverse order

Introduce a number: 1
Introduce a number: 2
Introduce a number: 3
Introduce a number: 4
Introduce a number: 0

The list of numbers is NOT the same as the list of numbers in reverse order
```

SOLUTION

```
vect = [];
count = 0;
numberRead = input('Introduce a number: ');
while (numberRead ~= 0)
    count = count + 1;
    vect(count) = numberRead;
    numberRead = input('Introduce a number: ');
  end

bReversible = 1;
bEnd = 0;
contMin = 1;
contMax = length(vect);
while ((bReversible) && (not(bEnd))) % same as while ((bReversible ==
1) && (bEnd == 0))
    if (vect(contMin) ~= vect(contMax))
        bReversible = 0;
    else
        contMin = contMin + 1;
        contMax = contMax - 1;
        if (contMin >= contMax)
            bEnd = 1;
        end
    end
end

if (bReversible) % same as if (bReversible == 1)
    disp('The list of numbers is the same as the numbers in reverse
order');
else
    disp('The list of numbers is NOT the same as the numbers in
reverse order');
end
```

Extra exercise 4

Write a program that asks the user to introduce a number and converts it to its binary base.

How to convert a number to its binary base: Divide the number you want to convert by 2 and write down the quotient and remainder of this division. Recursively divide the new quotients by 2 – taking note of the remainders – until the quotient is 1. The binary of the number will be 1, followed by the remainders of all the previous divisions, ordered from last to first.

Tip: you can use a vector to store the values of the binary number.

Example:

```
19 |_2
 1   9 |_2
     1   4 |_2
         0   2 |_2
             0   1
```

The binary of 19 is 1 0 0 1 1
Example of execution:

```
Introduce a number: 21
The binary is: 1 0 1 0 1
```

SOLUTION

```
clear;
divident = input('Introduce a number: ');
remainder = rem(divident,2);
quotient = floor(divident/2);
binary = [remainder]; % vector that will store remainders

while (quotient ~= 1)
    divident = quotient;
    remainder = rem(divident,2);
    quotient = floor(divident/2);
    binary = [remainder binary]; % concatenation
end
binary = [1 binary]; % 1 is always first number in binary number

fprintf('The binary is ');
for i = binary
    fprintf('%d ',i);
end
```