STAT380: Assignment 1

Ash Midgley

March 15, 2016

Question 1

For this question I set variables y and n to 1 and 10. I also set a result variable which kept track of the running total. I used a for loop to iterate through the values from 1 to n and added the equation total to the running total in the result variable.

When y was set to exp(1), R gave the output: [1]15.15233 Which agrees to at least 5 decimal places

R implementation:

```
y=1
n=10
result = 1


for(i in 1:n){
        result = result + (y^i)/factorial(i)
}


result

## [1] 2.718282
```

Question 2

For this question I set variables: x: The input vector values i: To keep track of the index in the vector len: To hold the length of the vector result: The output vector that is filled with the vector of moving averages.

R implementation:

```
x = c(1,2,3,4,5,6,5,4,3,2,1)
i=0
len = length(x)
result = c(length=(len-2))

while(i<(len-1)){
        result[i] = (x[i]+x[i+1]+x[i+2])/3
        i=i+1
}

result

##   length
## 2.000000 3.000000 4.000000 5.000000 5.333333 5.000000 4.000000
##
## 3.000000 2.000000
```

Question 3

The function for this exercise takes 1 argument m which is the input matrix.I set both the output matrices (mat1 and mat2) to the same value of the input matrix m. I then iterate through the matrix m. For each odd number, I have an if statement that doubles the value at the index in mat1. For each absolute value of less than 5 I use antoher if statement to set the value at that index to NA (negate the value).

R implementation:

```
matrix_adjust = function(m){
        size = length(m)
        mat1 = m
        mat2 = m
        for(i in 1:nrow(m)){
                for(k in 1:ncol(m)){
                        if((m[i, k]%%2) == 1){
                        mat1[i,k] = m[i,k]*2
                }
                if(abs(m[i,k]) < 5){
                        mat2[i,k] = NA
                }
        }
}
result = list(mat1, mat2)
return(result)
}

vec = c(3,7,8,12,-2,-3,19,4,5,5,6,3);
mat = matrix(vec, nrow=3,ncol=4,byrow=TRUE)

matrix_adjust(mat)

## [[1]]
##      [,1] [,2] [,3] [,4]
```

```
## [1,]    6   14    8   12
## [2,]   -2   -6   38    4
## [3,]   10   10    6    6
##
## [[2]]
##      [,1] [,2] [,3] [,4]
## [1,]   NA    7    8   12
## [2,]   NA   NA   19   NA
## [3,]    5    5    6   NA
```

Question 4

For this exercise I made a function with the name game. The function takes 1 argument x which is a positive int value. I used a while loop to keep processing the x value until it was equal to 1. I have a varaible steps that keeps track of the number of steps. I also have a variable max which keeps track of the maximum value that is reached.

steps begins with value 0 and for each repeat of the while loop it is incremented by 1. max begins with the value of x and I use a if statement in the while loop to check if the new x value is greater than the current max value. If it is larger, max becomes x.

R implementation

```
game = function(x){
        steps=0
        max=x
        while(x != 1){
                if(x > max){
                        max = x
                }
          if(x %% 2 == 1){
                        x = x*3+1
                }else{
                x = x/2
                }
                steps = steps+1
        }
        result = c(steps, max)
        return(result)
}

game(6)

## [1]  8 16

game(73)
```

```
## [1]  115 9232

game(74)

## [1]  22 112
```