Problem Set 1

Tate Mason

2025-01-22

1 Problem 1:

Use the stringr package and its `str_length` function to calculate the number of characters in each element of `x`.

```
library(stringr)
x <- c('economics', 'econometrics','ECON 4750')
str_length(x)
```

```
[1]  9 12  9
```

2 Problem 2:

Try three approaches to calculate the sum of the numbers 1 to n

2.1 Approach 1:

```
sum_one_to_n_1 <- function(n) {
    x <- seq(1:n)
    sum(x)
}
sum_one_to_n_1(100)
```

```
[1] 5050
```

2.2 Approach 2:

```
sum_one_to_n_2 <- function(n) {
    y <- (n*(n+1))/2
    y
}
sum_one_to_n_2(100)
```

```
[1] 5050
```

2.3 Approach 3:

```
sum_one_to_n_3 <- function(n) {
    sum <- 0
    for (x in 1:n) {
        sum <- sum + x
    }
    sum
}
sum_one_to_n_3(100)
```

```
[1] 5050
```

3 Problem 3:

3.1 Part A:

Write a function which computes the Fibonacci sequence.

```
fibonacci <- function(n) {
    a <- 0
    b <- 1
    for (i in 3:n) {
        c <- a+b
        a <- b
        b <- c
    }
```

```
    c
}
fibonacci(5)
```

```
[1] 3
```

3.2 Part B:

Write a function which computes the

```
alt_seq <- function(a,b,n) {
    for (i in 3:n) {
        c <- a+b
        a <- b
        b <- c
    }
    c
}
alt_seq(3,7,4)
```

```
[1] 17
```

4 Problem 4:

4.1 Part A:

Write a function which takes `x` as an argument and returns `TRUE` if prime or `FALSE` otherwise.

```
is_prime <- function(x) {
    if (x==2) {
        return(TRUE)
    }
    if (x <= 1) {
        return(FALSE)
    }
    for (i in 2:(x-1)) {
        if (x %% i ==0) {
            return(FALSE)
        }
```

```
    }
    return(TRUE)
}
a1 <- is_prime(7)
a2 <- is_prime(10)
print(c(a1,a2))
```

```
[1]  TRUE FALSE
```

4.2 Part B:

Write a function to list all prime numbers 1-n

```
prime <- function(n) {
    if (n>=2) {
        x = seq(2,n)
        primes = c()
        for (i in seq(2,n)) {
            if (any(x == i)) {
                primes = c(primes, i)
                x = c(x[(x %% i) != 0], i)
            }
        }
        return(primes)
    }
    else {
        return("Input should be at least 2")
    }
}
prime(100)
```

```
 [1]  2  3  5  7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

5 Problem 5:

5.1 Part A:

Counting observations in `Iris`

```
length(iris)
```

```
[1] 5
```

5.2 Part B:

Finding the mean sepal length in the dataset

```
mean(iris$Sepal.Length)
```

```
[1] 5.843333
```

5.3 Part C:

Calculate the average of the variable `Sepal.Width`. Package `dplyr` used but loading now shown.

```
iris %>% filter(Species == 'setosa') %>% summarise(mean(iris$Sepal.Width))
```

```
  mean(iris$Sepal.Width)
1               3.057333
```

5.4 Part D:

Sort the dataset by variable `Petal.Length` and print only the first ten rows

```
iris_sorted <- iris %>% arrange(Petal.Length)
head(iris_sorted, 10)
```

```
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1           4.6         3.6          1.0         0.2  setosa
2           4.3         3.0          1.1         0.1  setosa
3           5.8         4.0          1.2         0.2  setosa
4           5.0         3.2          1.2         0.2  setosa
5           4.7         3.2          1.3         0.2  setosa
6           5.4         3.9          1.3         0.4  setosa
7           5.5         3.5          1.3         0.2  setosa
8           4.4         3.0          1.3         0.2  setosa
9           5.0         3.5          1.3         0.3  setosa
10          4.5         2.3          1.3         0.3  setosa
```

6 Problem 6:

Create a function which solves the quadratic equation and provides two solutions.

```
quadratic_solver <- function(a,b,c) {
    p <- (-b + sqrt(b^2 - 4*a*c))/(2*a)
    m <- (-b - sqrt(b^2 - 4*a*c))/(2*a)
    print(c(p,m))
}
quadratic_solver(1,4,3)
```

```
[1] -1 -3
```