S.koushik

192121150

R programming

Assignment 2

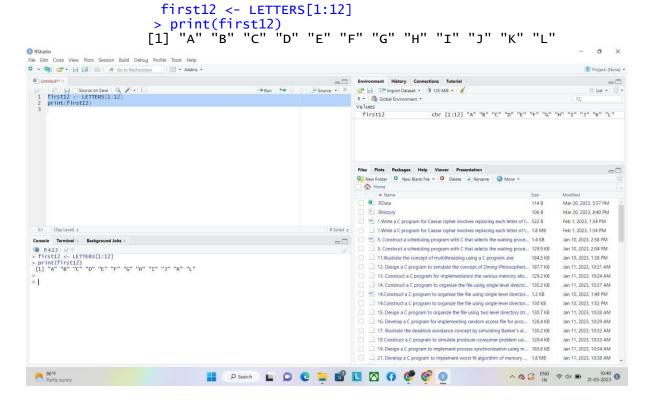
- 1. The built-in vector LETTERS contains the uppercase letters of the alphabet. Produce a vector of
 - (i) the first 12 letters;
 - (ii) the odd 'numbered' letters; (iii) the (English) consonants.

Program:

(i) the first 12 letters;

first12 <- LETTERS[1:12]

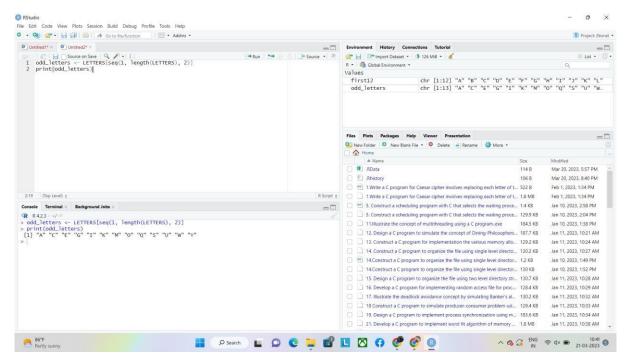
print(first12) output:



(ii) the odd 'numbered' letters; Program:
odd_letters <- LETTERS[seq(1, length(LETTERS), 2)] print(odd_letters)</pre>

Output:

```
odd_letters <- LETTERS[seq(1, length(LETTERS), 2)]
> print(odd_letters)
[1] "A" "C" "E" "G" "I" "K" "M" "O" "Q" "S" "U" "W" "Y"
```



(iii) the (English) consonants.

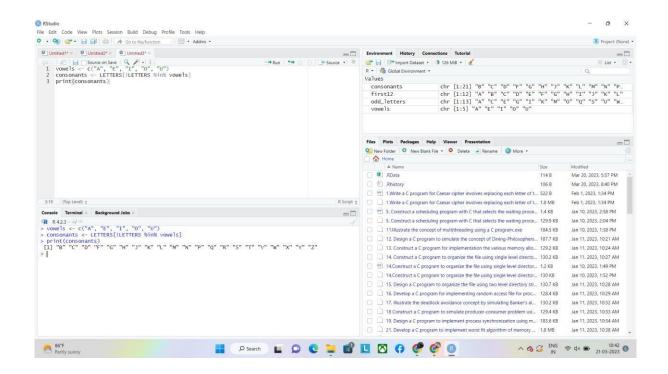
Program:

vowels <- c("A", "E", "I", "O", "U")

consonants <- LETTERS[!LETTERS %in% vowels]

print(consonants) output:

```
> vowels <- c("A", "E", "I", "O", "U")
> consonants <- LETTERS[!LETTERS %in% vowels]
> print(consonants)
  [1] "B" "C" "D" "F" "G" "H" "J" "K" "L" "M" "N" "P" "Q" "R" "S" "T" "V" "W" "X" "Y" "Z
```



2. The function rnorm() generates normal random variables. For instance, rnorm(10) gives a vector

of 10 i.i.d. standard normals. Generate 20 standard normals, and store them as x. Then obtain subvectors of

- (i) the entries in x which are less than 1;
- (ii) the entries between 0.5 and 1;
- (iii) the entries whose absolute value is larger than 1.5.

Program:

(i) the entries in x which are less than 1;

set seed for reproducibility set.seed(123)

generate 20 standard normals x

<- rnorm(20)

get subvector of entries in x which are less than 1
less_than_1 <- x[x < 1] print(less_than_1).</pre>

Output:

```
> # set seed for reproducibility
> set.seed(123)
>
> # generate 20 standard normals
> x <- rnorm(20)
>
> # get subvector of entries in x which are less than 1
> less_than_1 <- x[x < 1]
> print(less_than_1)
[1] -0.56047565 -0.23017749  0.07050839  0.12928774  0.46091621 -1.26506123 -
0.6868528
[8] -0.44566197  0.35981383  0.40077145  0.11068272 -0.55584113  0.49785048 -1.966617
[15]  0.70135590 -0.47279141
```

0 X O - 🧠 😅 - 🔒 🔒 👛 | 🖈 Go to file/function Source on Save Q / I # set seed for reproducibility 2 set.seed(123) 4 # generate 20 standard normals
5 x <- rnorm(20)</pre> 8 9 # get subvector of entries between -0.5 and 1 10 between_neg05_pos1 <- x[x > -0.5 & x < 1] 11 print(between_neg05_pos1) tw Folder | © New Blank File - | © Delete | E Rename | @ More • Home | A Name Mar 20, 2023, 5:57 PM Rhistory 106 B Mar 20, 2023, 8:40 PM 13:1 (Top Level) : 1.Write a C program for Caesar cipher involves replacing each letter of t... 1.8 MB Feb 1, 2023, 1:34 PM 5. Construct a scheduling program with C that selects the waiting proce... 1.4 KB R R4.23 · -/⇒
> # set seed for reproducibility
> set.seed(123) 5. Construct a scheduling program with C that selects the waiting proce... 129.9 KB Jan 10, 2023, 2:04 PM 184.5 KB 12. Design a C program to simulate the concept of Dining-Philosophers... 187.7 KB Jan 10, 2023, 1:38 PM Jan 11, 2023, 10:21 AM # generate 20 standard normals
> x <- rnorm(20)</pre> 13. Construct a C program for implementation the various memory allo... 129.2 KB Jan 11, 2023, 10:24 AM 14. Construct a C program to organize the file using single level directo... 130.2 KB 14.Construct a C program to organize the file using single level director... 12 KB Jan 10, 2023, 1:49 PM 15. Design a C program to organize the file using two level directory str... 130.7 KB Jan 11, 2023, 10:28 AM > Detweet_negus_post <- x(x > -0.3 & x < 1) print(between_neg05_post = 0.10 [1] -0.23017749 0.07050839 0.12928774 0.46091621 -0.44566197 0.35981383 0.40077145 [8] 0.11068272 0.49785048 0.70135590 -0.47279141 16. Develop a C program for implementing random access file for proc... 128.4 KB Jan 11, 2023, 10:29 AM 17. Illustrate the deadlock avoidance concept by simulating Banker's al... 130.2 KB Jan 11, 2023, 10:32 AM 18 Construct a C program to simulate producer-consumer problem usi... 129.4 KB Jan 11, 2023, 10:33 AM 19. Design a C program to implement process synchronization using m.... 183.6 KB 21. Develop a C program to implement worst fit algorithm of memory ... 1.8 MB Jan 11, 2023, 10:38 AM 86°F Partly sunny

Program:

(ii) the entries between – 0.5 and 1;

set seed for reproducibility set.seed(123)

generate 20 standard normals x

<- rnorm(20)

get subvector of entries between -0.5 and 1
between_neg05_pos1 <- x[x > -0.5 & x < 1]
print(between_neg05_pos1) output:</pre>

```
> # set seed for reproducibility
> set.seed(123)
> # generate 20 standard normals
> x <- rnorm(20)
> # get subvector of entries between -0.5 and 1
> between_neg05_pos1 <- x[x > -0.5 \& x < 1]
> print(between_neg05_pos1)
   \begin{bmatrix} 1 \end{bmatrix} - 0.23017749 \quad 0.07050839 \quad 0.12928774 \quad 0.46091621 \quad -0.44566197 \quad 0.359813 
83 0.40077145
   [8] 0.11068272 0.49785048 0.70135590 -0.47279141 >
                                                                                                                                                                    o x
  O • 🖎 😅 • 🔒 🔒 👛 📝 A Go to file/function
                                        - Addins -
                                                                                                                                                                 Project: (None) •
                                                                O Untitled1* × O Untitled2* × O Untitled3* × O Untitled4* × O Untitled5* × O Untitled6* ×
     1 # set seed for reproducibility
2 set.seed(123)
     4 # generate 20 standard normals
5 x <- rnorm(20)
    9 # get subvector of entries whose absolute value is larger than 1.5 10 abs_larger_than_15 < x[abs(x) > 1.5] 11 print(abs_larger_than_15)
                                                                                                 .RData
                                                                                                                                                  114 B
                                                                                                                                                            Mar 20, 2023, 5:57 PM
                                                                                                                                                           Mar 20, 2023, 8:40 PM
    11:26 (Top Level) :
                                                                                                    1.Write a C program for Caesar cipher involves replacing each letter of t... 1.8 MB
                                                                                                                                                           Feb 1, 2023, 1:34 PM
                                                                                       5. Construct a scheduling program with C that selects the waiting proce... 1.4 KB
   Console Terminal × Background Jobs ×
   R R4.23 · -/ □
> # set seed for reproducibility
> set.seed(123)

    5. Construct a scheduling program with C that selects the waiting proce... 129.9 KB

                                                                                                                                                           Jan 10, 2023, 2:04 PM

    12. Design a C program to simulate the concept of Dining-Philosophers... 187.7 KB

                                                                                                                                                           Jan 11, 2023, 10:21 AM

    13. Construct a C program for implementation the various memory allo... 129.2 KB
    14. Construct a C program to organize the file using single level directo... 130.2 KB
    # generate 20 standard normals
x <- rnorm(20)</pre>
                                                                                                                                                           Jan 11, 2023, 10:24 AM
                                                                                                                                                           Jan 11, 2023, 10:27 AM
                                                                                                 14.Construct a C program to organize the file using single level director... 1.2 KB
                                                                                                                                                           Jan 10, 2023, 1:49 PM
                                                                                                    14.Construct a C program to organize the file using single level director... 130 KB
   15. Design a C program to organize the file using two level directory str... 130.7 KB
                                                                                                                                                           Jan 11, 2023, 10:28 AM
                                                                                                    17. Illustrate the deadlock avoidance concept by simulating Banker's al... 130.2 KB
                                                                                                                                                           Jan 11, 2023, 10:32 AM
                                                                                                                                                           Jan 11, 2023, 10:33 AM
                                                                                                    19. Design a C program to implement process synchronization using m.... 183.6 KB Jan 11, 2023, 10:34 AM
                                                                                                 21. Develop a C program to implement worst fit algorithm of memory... 1.8 MB
                                                                                                                                                          Jan 11, 2023, 10:38 AM
                                                     86°F
Partly sunny
```

Program:

(iii) the entries whose absolute value is larger than 1.5.

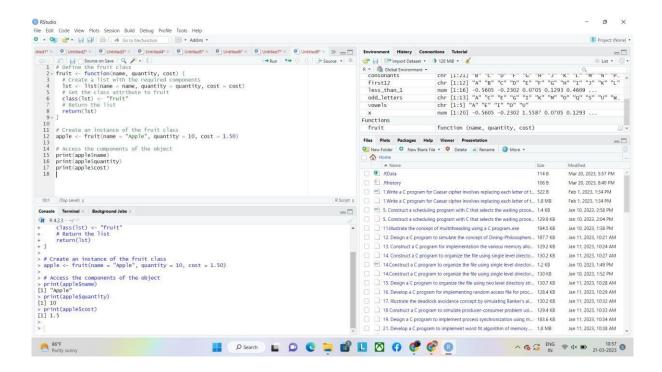
set seed for reproducibility set.seed(123)

generate 20 standard normals x

<- rnorm(20)

get subvector of entries whose absolute value is larger than 1.5
abs_larger_than_15 <- x[abs(x) > 1.5] print(abs_larger_than_15)
output:

```
> # set seed for reproducibility
> set.seed(123)
>
> # generate 20 standard normals
> x <- rnorm(20)
>
>
> # get subvector of entries whose absolute value is larger than 1.5
> abs_larger_than_15 <- x[abs(x) > 1.5] >
print(abs_larger_than_15)
[1] 1.558708 1.715065 1.786913 -1.966617
```



3. Solve the following system of simultaneous equations using matrix methods.

```
4a + 5b + c + 2d + 3e = 10 5a
+ b + 2c + 3d + 4e = 11
```

Program:

Define the matrix A and vector b

```
A <- matrix(c(1, 2, 3, 4, 5,

2, 3, 4, 5, 1,

3, 4, 5, 1, 2,

4, 5, 1, 2, 3,

5, 1, 2, 3, 4), nrow = 5, byrow = TRUE) b

<- c(-5, 2, 5, 10, 11)
```

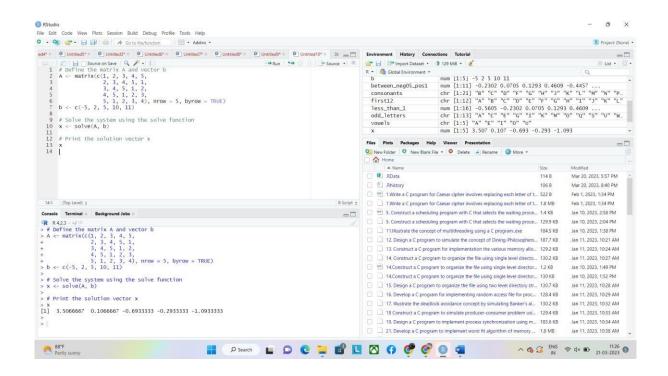
Solve the system using the solve function x

```
<- solve(A, b)
```

Print the solution vector x

X

Output:



4. Create a factor object for an apple color such as 'green', 'green', 'red', 'red', 'red', 'red', 'red', 'red'

green'. Print the factor and applying the nlevels function to know the number of distinct values

program:

create the factor object

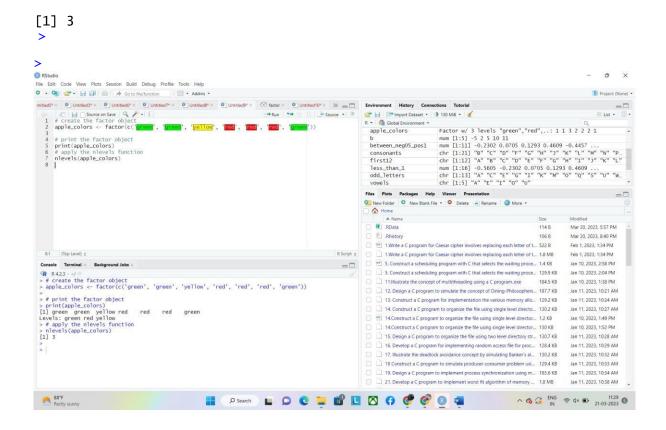
apple colors <- factor(c('green', 'green', 'yellow', 'red', 'red', 'green'))

print the factor object print(apple_colors)

apply the nlevels function

nlevels(apple colors) output:

```
> # create the factor object
> apple_colors <- factor(c('green', 'green', 'yellow', 'red', 'red', 'red', 'green'))
> 
> # print the factor object
> print(apple_colors)
[1] green green yellow red red green
Levels: green red yellow
> # apply the nlevels function
> nlevels(apple_colors)
```



5. Create an S3 object of class fruit contains a list with following required components such as name, quantity, cost and also Define and create s4 objects. Define a reference class of fruit program:

```
# Define the fruit class
```

```
fruit <- function(name, quantity, cost) { # Create a list
with the required components | lst <- list(name = name,
quantity = quantity, cost = cost)

# Set the class attribute to fruit
class(lst) <- "fruit" # Return the
list return(lst)
}</pre>
```

Create an instance of the fruit class
apple <- fruit(name = "Apple", quantity = 10, cost = 1.50)</pre>

Access the components of the object print(apple\$name) print(apple\$quantity) print(apple\$cost)

output:

```
> # Define the fruit class
> fruit <- function(name, quantity, cost) {</pre>
      # Create a list with the required components
      1st <- list(name = name, quantity = quantity, cost = cost)</pre>
      # Set the class attribute to fruit
class(lst) <- "fruit"</pre>
       # Return the list
      return(lst)
+ }
> # Create an instance of the fruit class
> apple <- fruit(name = "Apple", quantity = 10, cost = 1.50)</pre>
> # Access the components of the object
> print(apple$name)
[1] "Apple"
> print(apple$quantity)
[1] 10
> print(apple$cost)
[1] 1.5
```

