Applied Statistical Programming - Control Flow

Messi Lee, Alma Velazquez, Jordan Duffin Wong

1/31/2022

Write the R code to answer the following questions. Write the code and then show what the computer returns when that code is run. Make sure to change the appropriate header in the R code block to make this document compile.

You have until the beginning of class 2/2 at 10:00am to answer all of the questions below. You may use R, but not any online documentation. Submit the Rmarkdown and the knitted PDF to Canvas. Only one member of your group needs to submit the in class exercise.

1. Write code that will print every third number in $[1, 100]$.

```r
# Code
rm(list = ls())

for (i in 1:100) {
    if (i%%3 == 0) {
        print(i)
    }
}
```

```
## [1] 3
## [1] 6
## [1] 9
## [1] 12
## [1] 15
## [1] 18
## [1] 21
## [1] 24
## [1] 27
## [1] 30
## [1] 33
## [1] 36
## [1] 39
## [1] 42
## [1] 45
## [1] 48
## [1] 51
## [1] 54
## [1] 57
## [1] 60
## [1] 63
## [1] 66
## [1] 69
## [1] 72
## [1] 75
## [1] 78
## [1] 81
## [1] 84
## [1] 87
## [1] 90
## [1] 93
## [1] 96
## [1] 99
```

2. Write code that will print the following pattern in the R console.

1

1 2

1 2 3

. . . up to 20

. . . and then back down again

1 2 3

1 2

1

```r
# Code
rm(list = ls())


x <- rep(NULL, 20)

for (i in 1:20) {
    x[i] <- i
    print(x)
    if (i == 20) {
        for (j in 20:2) {
            x <- x[-j]
            print(x)
        }
    }
}
```

```
## [1] 1
## [1] 1 2
## [1] 1 2 3
## [1] 1 2 3 4
## [1] 1 2 3 4 5
## [1] 1 2 3 4 5 6
## [1] 1 2 3 4 5 6 7
## [1] 1 2 3 4 5 6 7 8
## [1] 1 2 3 4 5 6 7 8 9
##  [1]  1  2  3  4  5  6  7  8  9 10
##  [1]  1  2  3  4  5  6  7  8  9 10 11
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12
##  [1]  1  2  3  4  5  6  7  8  9 10 11
##  [1]  1  2  3  4  5  6  7  8  9 10
## [1] 1 2 3 4 5 6 7 8 9
## [1] 1 2 3 4 5 6 7 8
## [1] 1 2 3 4 5 6 7
## [1] 1 2 3 4 5 6
## [1] 1 2 3 4 5
## [1] 1 2 3 4
## [1] 1 2 3
## [1] 1 2
## [1] 1
```

3. Flip a coin (sample "*H*" and "*T*"). Have the script print the result of the coin flip into the R console until you get the same result three times in a row.

```
# Code
rm(list = ls())


coin <- c("H", "T")

flip <- function(x) {
    result <- sample(x, size = 1)
    return(result)
}

tosses <- NULL
stopper <- FALSE
set.seed(665)
while (stopper == FALSE) {
    tosses <- c(tosses, flip(coin))

    if (length(tosses) >= 3 & length(unique(tail(tosses, n = 3))) == 1) {
        stopper <- TRUE
    }

    print(tosses)
}
```

```
## [1] "T"
## [1] "T" "H"
## [1] "T" "H" "H"
## [1] "T" "H" "H" "H"
```

4. For each element of the following vector, print ‘True’ the number is positive and ‘False’ otherwise. `c(-1, -2, -3, 4, 5, 6, 7, -8, -9, -10)`

```r
# Code
rm(list = ls())


v <- c(-1, -2, -3, 4, 5, 6, 7, -8, -9, -10)

for (i in v) {
    if (i < 0) {
        print(FALSE)
    } else {
        print(TRUE)
    }
}
```

```
## [1] FALSE
## [1] FALSE
## [1] FALSE
## [1] TRUE
## [1] TRUE
## [1] TRUE
## [1] TRUE
## [1] FALSE
## [1] FALSE
## [1] FALSE
```

5. We know from the math modeling course that the sequence $\left\{\frac{1}{2^n}\right\}_{n=1}^{\infty}$ converges. Write a **`while`** loop that calculates what this sequence convergences to and keep track of the current and previous iterations. Have the **`while`** loop exit if the difference between iterations n and $n-1$ is less than 0.001. Upon exiting the loop, report the final value of the sequence.

```r
# Code
rm(list = ls())

set.seed(444)

series <- function(x) {
    term <- 1/(2^x)
    return(term)
}

n <- 1
stopper <- FALSE

while (!stopper) {

    diff <- abs(series(n) - series(n + 1))

    if (diff < 0.001) {
        stopper <- TRUE
        print(paste0("When n=", n, " terms are ~", round(diff, 5), " apart"))
    } else {
        n <- n + 1
    }



}
```

```
## [1] "When n=9 terms are ~0.00098 apart"
```