Contents

1 Programming Language Statements

A long list of Programming statements are as follows:

- `for`
- `while`
- `if`
- `break`
- `switch`

Due to time constraints, we will mainly look at the first two.

1.1 For Loops

For Loops: repeat the same set of instructions for a specified number of iterations. Each iteration is indexed (here with a value i).In the following example, we use the `cat()` command, whcih prints output to screen. (very useful in understanding loops).

```
for (i in 1:5)
    {
    # \n is the new line operator
    cat("This is loop",i, "\n")
    cat("The square of ", i, "is ",i^2,"\n\n")
    }
```

The output looks like this:

```
> for (i in 1:5)
+     {
+     cat("This is loop",i, "\n")
+     cat("The square of ", i, "is ",i^2,"\n\n")
+     }
```

```
This is loop 1
The square of  1 is  1

This is loop 2
The square of  2 is  4

This is loop 3
The square of  3 is  9

This is loop 4
The square of  4 is  16

This is loop 5
The square of  5 is  25
```

1.2 While Loops

While loops are useful when you dont know how many iterations are required in advance. The loop terminates when some logical condition has been meet. If the opposite logical condition is true, then the loop repeats again. There is no indexing necessarily, but we will use it in the following example.

In this example, we are trying to determine 5 numbers, randomly selected between 1 and 20, such that the mean of those five numbers is an integer.

```
#initialisation
X=sample(1:20,5)
i=0

while( mean(X) != floor(mean(X)) )
 {

 cat("This is attempt",i,"\n")
 cat("The mean value of X is",mean(X),"\n \n")

 X=sample(1:20,6)
 i = i+1
 }
```

The output should look like this:

```
...

This is attempt 10
The mean value of X is 9.166667

This is attempt 11
The mean value of X is 8.833333

This is attempt 12
The mean value of X is 12.33333

This is attempt 13
The mean value of X is 10.16667
```

We can print out all the relevant information with the following code

```
#print out the output
cat(
 "First Successful Attempt \n
 This is attempt",i,"\n
 The data set is", X , "\n
 The mean value of X is",mean(X),"\n")
```

```
First Successful Attempt

 This is attempt 14

 The data set is 10 15 4 17 18 2

 The mean value of X is 11
```

1.3 The Clever Lecturer

When writing an examination paper, a lecturer is trying to determine a simple data set of integers, such that the sample mean and sample standard deviation are also integers. This will make the examination easier to correct. In her data set, she decides to use a set of seven unsorted two-digit positive integer values. The following piece of code will be used.

```
SampSet = sample(10:99,7)
mean(SampSet)
sd(SampSet)
```

1.3.1 A First Attempt

A first attempt should look like this. However the mean and standard deviation are not integers. To check if a function is an integer, we can use the command `is.integer()`. (Another approach is using the `floor()`.function. A numeric value is an integer if it is equal to it's own floor function value.)

```
> SampSet
[1] 63 22 17 54 47 55 1
>
> mean(SampSet)
[1] 39.14286
>
> sd(SampSet)
[1] 20.09501
>
```

```
> is.integer(mean(SampSet))
[1] FALSE
>
> sd(SampSet)==floor(sd(SampSet))
[1] FALSE
```

The lecturer decides to implement a `while` loop to repeatedly sample 7 two-digit values until the statistics have integer values.

Necessarily if the standard deviation is an integer , necessarily the mean and variance must be integers also. This is very easy to prove, but we will not do so.

A logical condition must be specified to the while loop. If the standard deviation is not an integer, the loop will repeat. Once the standard deviation is an integer, the loop will stop.

```
SampSet = sample(10:99,7)


while( sd(SampSet)!=floor(sd(SampSet)) )
 {
 SampSet = sample(10:99,7)
 }
```

```
> SampSet
[1] 81 22 37 67 35 65 50
> mean(SampSet)
[1] 51
> sd(SampSet)
[1] 21
```

Suppose we wish to know how many loops were required to fulfil the condition. We would add a counter, that gets incremented at each iteration.

```
SampSet = sample(10:99,7)
Counter = 0


while( sd(SampSet)!=floor(sd(SampSet)) )
 {
 SampSet = sample(10:99,7)
 Counter = Counter+1
 }
```

1.4 Nested For Loops

A nested FOR loops is simply one FOR loop nested inside another, and is quite useful for performing operations on a two dimensional grid or matrix.

Let us specify a 6×6 dimensional array, called ***TwoDice***, which has as its elements the sum of the row and column numbers. (This is used to represent the sample space (i.e. each possible outcome) from rolls of two dice.

$$\text{TwoDice}[i,j] = i + j$$

```
# 6 sided dice
Die = 6

#prepare the output
TwoDice = matrix(0,nrow=Die,ncol=Die)

# i is the index for rows
 for(i in 1:Die)
   {
   #j is the index for columns
   for(j in 1:Die)
      {
      #simply assign the sum of the row and column to that element.
      TwoDice[i,j] = i+j
      }
   }

#Print out all the possible values
TwoDice
```

Different summations have different probabilities. We can compute the **textitprobability distribution** using the following code:

```
table(TwoDice)
sum(table(TwoDice))

prob2Dice = table(TwoDice)/sum(table(TwoDice))

#round the output to 4 decimal places

round(prob2Dice ,4)
```

Exercise Compute the **Expected Value** of this experiment.

- `sum()` sum of items in a vector
- `dim()` dimensions of a data object.
- `prod()` product of items in a vector.

```
> sum(TwoDice)
[1] 252
> dim(TwoDice)
[1] 6 6
> prod(dim(TwoDice))
[1] 36
> sum(TwoDice)/prod(dim(TwoDice))
[1] 7
```

Exercise Suppose we are interested in the absolute difference of the values of two 8-sided dice. What is the probability distribution? What is the expected value?

1.5 Random Grids

(See Risk Lab Exhibition) Lets construct a ”chequerboard” of randomly selected colours? How random does this look?

```
# Select a 9 by 9 grid
M=9

#prepare the graphical output
plot(c(1, M+1), c(1, M+1), type= "n", xlab="", ylab="")

#Lets practice drawing a blue rectangle
rect(1, 1, 2, 2,col="blue")


# ColPal1 = c("red", "blue", "yellow", "white", "green", "black")

# Lets use this coluir pallette
ColPal2 = c("white", "pink", "red", "black")


for( i in 1:M)
   {
   for( j in 1:M)
      {
      #  For this grid element, draw a rectangle with a colour
      #  randomly selected from the pallette.
      rect(i, j, i+1, j+1,  col=sample(ColPal2,1))


      }
   }

#
```