COMP 364 - Tools for the Life Sciences

Midterm
Prof. M Hallett; TA M Ghadie
March 10th, 2016
20% of total grade

Question 1: [20 points] U R R.

For each of the sub-questions below, describe the output from R.

[2 points each]

a

```
(x <- 5)
 x <- 6
(x <- x*2)
```

b

```
2*(1:5)
```

c

```
if ( FALSE && ( (!TRUE | !!!!TRUE) | (!!!!FALSE | !!TRUE) ) ) {
     print("ya baby")
 } else {
     print("no baby")
 }
```

d

```
for (i in c("hey", "ho", "schmo", "flow") ) {
     if (i == 4) {
          print(i)
     }
}
```

e

```
x <- 7
myFunction <- function( x = 3 ) {
     x <- 6
     return(x*3)
}
x <- 8
print(myFunction(4))
print(x)
```

f

```
a <- LETTERS[1:5]
b <- 1:5
c <- TRUE
d <- b
e <- list( a, b, list(c), d )

e[4]
e[[4]]
```

g

```
e[2:4]
e[[2:4]]
```

h

```
myGrades <- factor( c("A", "F", "K", "B", "A", "K", "C") )
myGrades
```

i Consider the following function. Recall that the `sample(x=..., size=..., prob=..., replace=...)` function chooses elements from `x` randomly; where (1) `size` is number of times, (2) `replace` defines whether or not to select with replacement, and (3) the elements of `x` are chosen according to the probabilities in the vector `prob`. If `prob` is not specified then, by default, each element of `x` has the same probability of being selected. (i.e. if `prob` is not specified we select elements of `x` uniformly at random).

```
randomCoinFlips <- function(flips = 10, prob.heads = 1 ) {
          return( sample( x= c("H", "T"),
                          size=flips,
                          replace=TRUE,
                           prob=c(prob.heads, 1 - prob.heads)))

                }
```

Describe the output of the following:

```
randomCoinFlips()
```

j

```
aa <- LETTERS[1:5]
bb <- 1:5
cc <- c(TRUE, TRUE, FALSE, TRUE, FALSE)
dd <- data.frame( letterz = aa, numbs = bb, truthity = cc )
which(dd$truthity)
subset(dd, truthity)
```

Question 2: [20 points] Basic Programming Structures

Recall the following for manipulating strings and characters:

```
x <- "Hello"
substr(x, start=2, stop = 4)
                      # substring of between 2 and 4 (inclusive!)
> "ell"

paste(x, x, sep="")  # collapse a vector of strings, nothing between
> HelloHello

toupper(x)
> "HELLO"

nchar(x)        # nchar() returns the length of the string
> [1] 5
```

In this question, you should write **a single function** that finds the Watson-Crick (WC) complement of a given string (A<->T, C<->G). In particular, your function should meet the following conditions:

- It should be called "`convertToWC`"
- It should accept a parameter called "`target.dna`" No default value need be specified.
- It should accept a parameter called "`five.prime`". If `five.prime` is `TRUE`, your function should return the WC complement of `target.dna` in the reverse order. If `five.prime` is `FALSE`, then it should return the WC complement in the same direction. Default value is `TRUE`.
- If a character in `target.dna` is in lower-case, it should be switched to upper-case.
- If a character in `target.dna` is not A, C, G or T, then the output string should contain an X.
- Function `convertToWC` should return the WC-complement of `target.dna` in the correct direction (determined by `five.prime`) in uppercase.

Question 3: [20 points] Human Compendium (huc) of Gene Expression.

Recall that a single gene may have multiple probes. This information is in the huc data structure.

[2 points] Show how you would load the huc with the `vanvliet` dataset.

For the questions below, assume you have loaded the huc into a variable called `huc` with the `vanvliet` dataset, accessible as `huc$vanvliet`.

[8 points] Write R code that finds the gene(s) that has(have) the most probes in the `vanvliet` dataset. Print out the gene name(s). If there is a tie for the highest number of probes, make sure your code prints out all of the different gene names.

[10 points] Write R code that solves the following problem. For any gene that has exactly two probes in `vanvliet`, compute the absolute difference in expression between the two different expression measurements for each patient, and find which patient has the maximum absolute difference in expression.