CentralDogma

```
library(CentralDogma)
```

Group 11
Github repository: https://github.com/rforbiodatascience22/group_11_package

Description of the package

When you purify and sequence DNA, you may want to: * Transcribe the DNA sequence to RNA. * Get the codons corresponding to the translated RNA. * Analyse the frequency of the codons in your peptide.

The `CentralDogma` package allows you to perform such analysis and also offers the possibility to create a random DNA sequence to practice with its set of tools.

Creating a random DNA sequence with `randomDNASequence`

`randomDNASequence` takes the length of the DNA sequence as argument and returns a random selection of nucleotides.

For example, we can opt to create a DNA sequence comprised of 100 nucleotides and assign it to a variable:

```
dnaSequence <- randomDNASequence(100)
dnaSequence
#> [1] "CTTTGTGCACTCTACCTGACACGGTCCCAGCAGTTGGCTCTATTCCTCAGGAGGAGGGATTGTGATCATGGAGTGAGGTATGTTATATCGCACGC
```

Transcribing our DNA sequence with `DnaToRna`

After creating the random DNA sequence or obtaining it elsewhere, `DnaToRna` takes the variable containing the nucleotides as input and transcribes it to RNA.

```
rnaSequence <- DnaToRna(dnaSequence)
rnaSequence
#> [1] "CUUUGUGCACUCUACCUGACACGGUCCCAGCAGUUGGCUCUAUUCCUCAGGAGGAGGGAUUGUGAUCAUGGAGUGAGGUAUGUUAUAUCGCACGC
```

Detecting the codons in the RNA strand with `getCodons`

`getCodons` allows to divide the strand of RNA into its composing 3-nucleotide codons.

```
codonSequence <- getCodons(rnaSequence)
codonSequence
#>  [1] "CUU" "UGU" "GCA" "CUC" "UAC" "CUG" "ACA" "CGG" "UCC" "CAG" "CAG" "UUG" "GCU" "CUA" "UUC" "CUC"
#> [27] "UAU" "GUU" "AUA" "UCG" "CAC" "GCG" "CCG"
```

Translating the codon sequence into a peptide chain with `translateRNASequence`

The next step is translating the codon sequence to obtain the polypeptide chain, and this is possible with `translateRNASequence`.

Importantly, if your codonSequence variable contained a STOP codon, it will be represented with an asterisk * in the output.

```
peptideSequence <- translateRNASequence(codonSequence)
peptideSequence
#> [1] "LCALYLTRSQQLALFLRRRDCDHGVRYVISHAP"
```

`plotFrequencies` analyses the frequency of amino acids in youor polypeptide chain

Finally, it is often relevant to estimate the number of the different amino acids in order to understand the chemicophysical properties of your protein. `plotFrequencies` will generate a barplot to display this information.

Importantly, if your polypeptide chains contains a STOP (*), it should be remmoved before plotting the data.

```
#plotFrequencies(peptideSequence)
```

Addenda

Discussion of Task 4

1. Each function has been previously described.
2. Regarding the three packages as dependencies in function 5 (`plotFrequencies`), it was achieved by writing the following in the console:
   - `usethis::use_package("magrittr")`
   - `usethis::use_package("stringr")`
   - `usethis::use_package("ggplot2")`

   The pipe operator was included by writing `@importFrom magrittr %>%`
3. Including dependencies is necessary whenever the package uses functions contained in other packages. For example, this package uses functions from `ggplot2`, so its inclusion in the dependencies is necessary. However, it is a good idea to try and reduce the number of dependencies, as anyone using our package will be forced to download all of them. The more dependencies there are, the more time it will take for people to be able to use the package. Furthermore, if these dependencies have dependencies of their own, it will take even more time to load.
4. The difference between using `@importFrom package function` and `package::function()`is that the former allows to use the function repeatedly after importing, while the latter is for a one-time use.

Other functions that could be included in the package

A function could be created to discriminate the amino acids in the peptide sequence based on their charges. Then, this could be used in the `plotFrequencies`function to create a plot stratified by the nature of the amino acids present in our peptide.