Biostatistică și Bioinformatică.

Lab 1: Ilustrarea unor procese din biologia moleculară.
Accesarea bazelor de date biologice.
Familiarizare cu R - pachete pentru bioinformatică

I. Vizualizarea proceselor de transcriere și translație

- se accesează http://vcell.ndsu.nodak.edu/animations/
- se activează succesiv animațiile de la procese moleculare (Molecular Processes):
  - transcrierea ADN-ului în ARN (Transcription)
  - prelucrarea ARN-ului mesager (mRNA Processing)
  - secționarea ARN-ului mesager (mRNA Splicing)
  - translație – sinteza lanțului peptidic (Translation)

II. Accesarea resurselor de la NCBI – National Center for Biotechnology Information (http://www.ncbi.nlm.nih.gov)

Conține:
- baze de date biologice - GenBank - http://www.ncbi.nlm.nih.gov/genbank/
  - Nucleotide - http://www.ncbi.nlm.nih.gov/nucleotide/
  - Proteine - http://www.ncbi.nlm.nih.gov/protein
- instrumente software pentru analiza datelor biologice grupate într-un sistem de căutare și regăsire a informațiilor (Life Science Search Engine: Entrez - http://www.ncbi.nlm.nih.gov/gquery/gquery.fcgi)
- resurse referitoare la genomul uman
- resurse pentru documentare

Exercitii:

1. Vizualizarea structurii genomului uman.

   Indicatie: se accesează MapViewer (http://www.ncbi.nlm.nih.gov/mapview/);
   se alege la Search: Homo Sapiens;
   se selectează unul dintre cromozomi și se vizualizează secvența (View Sequence)

2. Să se localizeze toate prezențele subsecvențelor TATA respectiv TATAAAA (subsecvențe de tip promoter) în cromozomul 1 al genomului uman

   Indicatie. După accesarea lui MapViewer se specifică subsecvența în câmpule Search for și 1 la Chromosome

3. Să se salveze secvența corespunzătoare ADN-ului mitocondrial (conține informația genetica transmisă doar pe cale maternă și este prin urmare afectată doar de mutații) in formatele GenBank respectiv FASTA

   Indicație.
   (i) Conectare http://www.ncbi.nlm.nih.gov/mapview/
   (ii) Selectare Homo Sapiens

(iii) Selectare ADN mitochondrial (click pe MT)
(iv) Selectare GenBank sau FASTA
(v) Click pe Send + selectare salvare in fişier a întregii înregistrări

Observatie: formatul FASTA este cel mai simplu format de stocare a secvenţelor de nucleotide sau de aminoacizi. Un fisier FASTA conţine o linie antet (care începe cu simbolul > şi conţine o scurtă descriere a secvenţei) şi continuă cu linii conţinând câte 80 de simboluri (cu excepţia ultimei linii care poate fi mai scurtă). Liniile care încep cu simbolul ; sunt considerate comentarii.

III. Mediul R (http://cran.r-project.org/)

R este o suită integrată de instrumente software destinate prelucrării datelor, efectuării unor calcule şi vizualizării grafice. R conţine o colecţie mare de pachete destinate diferitelor tipuri de prelucrări (în principal de natura statistică) şi un limbaj care permite extinderea facilităţilor prin definirea de noi funcţii şi pachete.

Caracteristici ale limbajului R.

Aspecte generale:

- Este case-sensitive
- Cea mai simplă şi frecvent utilizată comandă (de atribuire) se specifică prin
  nume_ variabila <- valoare sau
  nume_ variabila = valoare sau
  assign("nume_variabila",valoare)
- Comenzile (instrucţiunile) se separă prin ; sau se plasează pe linii separate
- Comenzile elementare pot fi grupate prin specificare între { şi }
- Comentariile se specifică prin #
- Revenirea la o comandă anterioară se poate face cu "săgeată sus"
- Execuţia comenzilor stocate într-un fişier se face prin
  source("fisier_comenzi.R")

Obiecte în R:

- **Scalari** (valori individuale care pot fi interpretate ca vectori cu un singur element):
  - Valori numerice (întregi, reale, complexe)
    - Conversii între tipuri: var2=as.tip_nou(var1) (ex: as.character(2) returneaza "2")
  - Logice (constante predefinite: TRUE sau T, FALSE sau F)
    - Operatori logici: ! (negatie), | (disjunctie), & (conjunctie)
    - Operatori relationali: <,>,<=,>=,== (egalitate),!= (diferit)
  - Simboluri
  - Constante uzuale:
    - NA (valoare nedefinită)
    - Pi
- **Şiruri de caractere**: se specifică între ghilimele sau apostroafe.
  - Concatenarea şirurilor se poate realiza cu funcţia paste.
  - Conversii: c2s: vector de caractere -> sir de caractere; s2c: sir de caractere in vector de caractere

- Specificare subsir: substring(subsir, indiceStart, indiceFinal)
- Analiza:
  - words.pos(subsir,sir): determina toate pozițiile de start ale subșirului in șirul dat

- **Vectori** (colecții omogene și ordonate de valori accesibile prin indice de poziție).

  - Construire vector: nume_vector <- c(el1,el2,...,eln)
  - Accesare element: nume_vector[indice] (indicii sunt numerotați începând de la 1)
  - Operații:
    - operațiile aritmetice (+,-,*,/,^) și funcțiile matematice unare (exp, log, sqrt,sin,cos etc.) aplicate asupra vectorilor acționează la nivelul fiecărui element. Pentru operatorii binari nu e obligatoriu ca cei doi vectori să aibă același număr de elemente – rezultatul va avea lungimea celui mai lung vector iar operandul mai scurt va fi completat ciclic cu aceleași elemente. De exemplu a=c(1,2,3); b=c(4,1); c=a+b va conduce la rezultatul 5 3 7 (c este completat la (4,1,4))
    - determinarea numărului de elemente din vector: length(vector)
    - determinare minim respectiv maxim: min(vector) respectiv max(nume_vector)
    - calcul suma respectiv produs elemente: sum(vector) respectiv prod(vector)
    - sortare crescătoare: sort(vector)
    - selecție elemente: nume_vector[secventa_indici] (ex: vector[c(1,3,5)] returnează elementele de pe pozițiile 1, 3 și 5); secvența de indici poate fi creată prin orice funcție (ex: vector[vector>0] returnează vectorul conținând doar elementele pozitive din vectorul inițial.
    - Replicarea unui obiect: rep(obiect,times=nr_repetari). De exemplu pentru b=c(4,1) prin rep(b,times=3) se va genera 4 1 4 1 4 1

  - Generarea unor secvențe de valori:
    - Valori consecutive: valoare_initiala:valoare_finala (ex: 1:5 generează vectorul corespunzător lui c(1,2,3,4,5))
    - Valori separate printr-un pas specificat: folosind funcția seq(valoare_initiala, valoare_finala, by=pas) (ex: seq(1, 5, by=1))

- **Liste** (colecții nu neapărat omogene de obiecte accesibile prin indice sau nume)
  - Construire lista: lista=list(element1, element2,...); fiecare element al listei poate fi un obiect anonim sau cu nume
    (ex: lista1=list("zi"=23,"luna"="februarie","an"=2014))
  - Accesarea valorilor elementelor:
    - Prin indici: lista[[indice]] (ex: lista1[[1]] va returna 23) (Obs: dacă se folosește lista[indice] se returnează atât valoarea cât și numele elementului.
    - Prin nume: lista$nume_element (ex: lista1$an va returna 2014)
  - Concatenarea listelor: c(lista1,lista2,...)
  - Modificarea unei liste: lista[[indice]]=element_nou
- **Tablouri** (uni și multi-dimensionale)
  - Construire: tablou=array(vector_valori,dim=c(dim1,dim2, ...)
    ex: matrice=array(1:6,c(2,3))

- Specificare elemente: tablou[indice1,indice2,...] (Obs: există multe variante de a accesa subseturi de elemente aflate pe diferite poziții în matrice)
- Determinare dimensiuni matrici: nrow (numar de linii), ncol (numar de coloane)
- Combinare tablouri: cbind (alipire orizontala – coloana dupa coloana), rbind (alipire verticala – linie dupa linie)
- Operațiile aritmetice clasice (+,*,-,/) se pot aplica asupra unor tablouri fiind mapate la nivel de element (ca în cazul vectorilor). Pentru a calcula produsul a două matrici (calculând produsul scalar al liniilor şi coloanelor corespunzătoare) se poate folosi operatorul %*%

- **Tabele de date (data frame):** sunt tabele în care fiecare linie corespunde unei înregistrări şi fiecare coloană corespunde unei caracteristici. Valorile de pe aceeaşi coloană trebuie să fie de acelaşi tip dar cele de pe coloane diferite pot fi de tipuri diferite.
  - Construire: data.frame(nume_caracteristica1= valori_caracteristica1, nume_ caracteristica 2= valori_ caracteristica 2,...). Exemplu: df=data.frame(a1=c(1,2,3),a2=c("a","b",”c”))
  - Completare prin citire din fisier: read.table(“nume_fisier”). Fisierul trebuie să conțină pe prima linie numele caracteristicilor şi toate liniile trebuie să conțină acelaşi număr de valori
- **Tabele de frecvențe**
  - Construire: tabel_frecv=table(vector) (ex: table(c(2,1,3,1,2)))

Obs: obiectele din R au atribute specifice care pot fi vizualizate folosind funcția attributes şi modificate folosind attr. De exemplu prin attributes(df) se obțin valorile corespunzătoare pentru atributele: $names, $row.names şi $class iar prin attr(df,”names”)=c(“b1”,”b2”) se modifica numele caracteristicilor din “a1” si “a2” in “b1” respectiv “b2”.

Structuri de control

- Prelucrări decizionale: if (conditie) instructiune1 else instructiune 2
- Prelucrări repetitive:
  - **For:** for (contor in domeniu) instructiune
    (obs: in varianta cea mai simpla domeniul se specifica prin valoarea_initiala:valoarea_finala)
  - **While:** while(conditie) instructiune
  - **Repeat:** repeat instructiune (intreruperea ciclului repeat se poate face doar prin break).

Definirea funcțiilor

- Definirea unei funcții: nume_functie = function(arg1, arg2, …){instructiuni; return(rez)}
- Apelul unei funcții: nume_functie(val1,val2,….) (Obs: la apel în loc de a specifica doar valoarea unui argument (val1) se poate specifica o pereche de forma nume_argument=valoare (ex: arg1=val1). In acest caz ordinea parametrilor de apel nu trebuie să coincide neapărat cu ordinea specificată la definirea funcției.
- Particularități:
  - Variabilele utilizate în cadrul unei funcții sunt implicit considerate variabile locale astfel că efectele operațiilor de asignare specificate cu <- sau = au doar efect local

- Pentru a efectua o asignare cu caracter global (al cărui efect să rămână după revenirea din apel) se poate folosi operatorul de "superasignare": <<-

Exemplu definire funcție:

```
suma=function(v)
 {s<-0
  for (i in 1:length(v)) s<-s+v[i]
 return(s)
}
```

Exemplu de apel: suma(1:5) (funcția trebuie încărcată în prealabil folosind în fereastra de comandă: File->Source R code urmat de selectarea fișierului care conține definirea funcției)

Prelucrări grafice simple

- Graficul unei funcții: plot(nume_functie, limita_inferioara, limita_superioara) (ex: plot(sin,-2*pi,2*pi))
- Grafic liste puncte: plot(vector_abscise,vector_ordonate) (ex: plot(a,a^2) cu a un vector cu valori numerice)

Citirea datelor din fișiere

- Citire sub forma de obiecte de tip "data frame" (tabele de valori în care prima linie conține denumiri ale câmpurilor iar prima coloană conține numărul de ordine al înregistrării). Este adecvată pentru citirea din fișierele ce conțin înregistrări specificate pe câte o linie.
  - Apel: read.table("nume_fisier")

- Citire sub forma de vector cu elemente de tip vector (în cazul în care fișierul conține linii cu același număr de valori separate prin spații dar nu conține o linie cu antet). Un element al vectorului returnat corespunde unei coloane din fișier (rezultatul nu e o structură de tip matrice astfel ca pentru a accesa a treia valoare din linia 2 trebuie specificat rez[[3]][[2]].
  - Apel: scan("nume_fisier",lista_tipuri) (ex: scan("fisier.dat",list("",0)) – citește un fișier cu două coloane în care pe prima coloană se află caractere iar pe a doua valori numerice.

- Citire fișiere de tip FASTA. Fișierele de tip fasta pot fi citite folosind funcții din pachetul Seqinr (necesita instalarea prealabilă folosind facilitatea Package din meniu):
  - Incarcare pachet: library(seqinr)
  - Citire fișier: date=read.fasta("nume_fisier")
  - Extragerea secvenței ADN: secv=date[[1]] sau secv=getSequence(date)
  - Determinarea lungimii secvenței: length(secv)
  - Determinarea frecvenței de apariție a nucleotidelor: table(secv)

Exercitii:

1. Să se determine frecvența de apariție a fiecarui tip de nucleotidă din secvența aflată în fișierul mtDNA.fasta
2. Să se determine frecvența de apariție a fiecarui tip de aminoacid din secvența aflată în fișierul mtProtein.fasta

Indicație:

a) Se încarcă pachetul seqinr (prin Packages->Load Package; dacă nu este în listă trebuie instalat în prealabil prin Packages->Install Package)
b) Se citește fișierul corespunzător: secv=read.fasta("nume fisier")
c) Se construiește tabelul de frecvențe: table(secv[[1]])

3. Să se determine densitatea de perechi de forma GC dintr-o secvență.

Indicație: se localizează toate prezențele subșirului „gc" in sirul ce contine secventa: p=words.pos("gc",c2s(secv[[1]])). Pentru a determina densitatea este suficient să se calculeze raportul length(p)/length(secv[[1]]). Pentru a verifica rezultatul se poate folosi funcția count(secv[[1]],2) care determină numărul de apariții ale fiecăreia dintre cele 16 perechi posibile de nucleotide.

4. Să se determine toate subsecvențele care încep cu "gt" și se termină cu "ag" (ar corespunde porțiunilor de tip introni).
   Indicație: se identifică perechile "gt" si "ag" și se combină astfel încât pozițiile lor de start să fie intercalate (similar procesului de interclasare a listelor sortate).

5. Să se vizualizeze codul genetic.
   Indicație: se poate folosi funcția tablecode() din pachetul seqinr

Temă: Scrieți funcții (limbajul de programare este la alegere) pentru extragerea secvenței de nucleotide din fișiere în format FASTA respectiv GenBank.