PEiAR - Laboratorium 1 **26.02.2024**

1) Instalacja pakietu R (The R Project for Statistical Computing)
2) Instalacja dedykowanego IDE (Integrated Development Environment): RStudio (rstudio.com/products/rstudio/)

Zad.0 (Proste typy danych w R oraz symbole NA, NaN)

Zad.1 (R jako kalkulator)
Oblicz
a) $\log_2 1024$ //komenda log(…, …)// ln(2.718); $\log_5 625$; $\log_{10} 1000$ //komenda log10(...)//

b) sin(30°); cos(Π/4); e //komenda exp(...)//; e^7; √e; arctg(1) //komenda atan(...)//;

c) 4! //komenda factorial(...)//; 8 nad 2 – czyli symbol Newtona //komenda choose(…, …)//

d) zaokrąglij liczbę Π do 4 miejsc po przecinku tj. do 5 tzw. miejsc 'znaczących' //komenda signif(…, …); wyznacz podłogę i sufit liczby e; wyznacz wartość bezwzględną liczby (e^2 Π – 27)

e) wyznacz resztę z dzielenia 1286 *mod* 7; wyznacz część całkowitą z dzielenia 1286 *div* 7

f) wyznacz część rzeczywistą i urojoną liczby zespolonej o module 4 i argumencie 3/2Π //komenda Re(…) oraz Im(…) użyta dla liczby zespolonej wyznaczonej przy użyciu polecenia complex(modulus = …, argument = …) //; wyznacz moduł i argument liczb zespolonej -2 + 4i //komenda Mod(…) i Arg(…) //

Zad.2 (wektory)
a) wygeneruj wektor, którego współrzędnymi są liczby naturalne 1, …, 10 przy użyciu:
   i) c(…, …, …)
   ii) znaku dwukropka
   iii) seq(…, …, by = …)
b) wygeneruj wektor, którego współrzędne stanowią ciąg arytm. liczb od 12 do 104 zawierający 20 wyrazów //seq(…, …, length = …)//
c) wygeneruj wektor, którego współrzędne stanowią czterokrotnie powtarzający się ciąg liczb parzystych od 2 do 8 //rep(…, 4)//
d) wygeneruj wektor o współrzędnych 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 4 4 4 4 4 5 5 5 5 5 //rep(…, each = 5)//
e) wygeneruj wektor o współrzędnych 1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 8, 8, 8, 8, 8, 8, 8, 8

Zad.3 (indeksowanie wektorów)
a) wygeneruj i przechowaj pod zmienną a wektor, którego współrzędne stanowią ciąg liczb arytm. o wyrazie pierwszym 1, ostatnim 37 i różnicy r = 4; zwróć:
   i) jego drugą współrzędną //a[…]
   ii) wszystkie jego współrzędne oprócz trzeciej // a[- …]
   iii) wszystkie współrzędne oprócz pierwszej i dziesiątej
   iv) tylko te współrzędne, których wartość jest większa niż 18 //a[...warunek…]
   v) liczbę współrzędnych, które spełniają warunek z ppkt. (iv) // length(…)
   vi) współrzędne, których kwadrat jest liczbą parzystą
   Uwaga: numeric(0) – oznacza wektor o współrzędnych numerycznych (liczbowych) mający zerową długość
   vii) Co zwróci polecenie a > 18 ?

b) wygeneruj wektor *a*, którego współrzędne stanowią 26 pierwszych wyrazów ciągu geometrycznego o wyrazie pierwszym 2/5 i ilorazie q = 3 ;

i) policz średnią z danych w wektorze *a* // mean(…)
ii) co drugiej współrzędnej wygenerowanego wektora zmień znak na przeciwny (nowy wektor *b*)

Uwaga: Gdy mnożymy przez siebie wektory o różnych długościach, to po dojściu do końca krótszego z wektorów, obliczenia rozpoczynają się ponownie dla pozostałych współrzędnych wektora dłuższego => np. mnożąc wektor a = 1:7 przez wektor b = c(1, 1, 3) otrzymamy wektor: [1, 2, 9, 4, 5, 18, 7] //

iii) pierwszą, czwartą i ostatnią współrzędną wektora otrzymanego w (ii) zastąp liczbami 0, Π oraz e, odpowiednio (niech to będzie nowy wektor *d*)
iv) wygeneruj wektor *f*, którego współrzędne są odwrotnościami współrzędnych wektora *d*

Uwaga: funkcje w R definiujemy klasycznie: nazwaFunkcji = **function(…) { … }**
Co ważne – ciało funkcji kończy się metodą kluczową **return(…)**

Zad. 4 (ciąg Fibonacciego)
Napisz funkcję zwracającą zadaną w argumencie liczbę *n* pierwszych wyrazów ciągu Fibonacciego zadanego rekurencyjnie:

$a_0 = 0$,
$a_1 = 1$,
$a_n = a_{n-1} + a_{n-2}$ dla $n \geq 2$.

Wsk. możesz użyć np. pętli **<u>for</u> (…*element* ... <u>in</u> … *zbiór danych* ...) {… *instrukcja wykonana podczas każdej iteracji* ...}**

Zad. 5 (pętla while i kilka przydatnych funkcji…)
Wykorzystując pętlę **while (… *warunek* ...) {… i*nstrukcja wykonywana w przypadku pozytywnej weryfikacji warunku* …},** chcemy powtarzać losowanie 5 (docelowo dowolnej zadanej ilości) liczb naturalnych nie większych niż 100, do momentu, aż najmniejsza różnica między dowolnymi wylosowanymi (w danej iteracji) liczbami będzie mniejsza lub równa 2. Czyli jeśli np. w pewnej iteracji pętli, wylosowany zostanie zestaw:
3, 10, 54, 6, 76,
to losowanie musi zostać powtórzone ponieważ najmniejszą różnicą jest teraz 3 (pomiędzy liczbą 3 i liczbą 6) .
Wsk. użyj pomocniczych funkcji:

sample(górne ograniczenie dla losowanych liczb**,** liczebność losowanego zbioru **)** //losuje liczby naturalne nie większe od pierwszego argumentu//

diff(…) //zwraca wektor różnic między kolejnymi współrzędnymi wektora zadanego w argumencie//

sort(…) //porządkuje (domyślnie rosnąco); sort(…, decreasing = TRUE) → pozwala ustawić porządek malejący//

min(…) // komentarz chyba nie jest potrzebny