APPELLO 25 SETTEMBRE 2023

Domanda 1 (3 punti). Dopo aver scaricato il file swiss_data.RData dal sito upload, caricare in R i dati ivi contenuti ed esplorare cosa è stato aggiunto al workspace. (2 punti). Calcolare anche classe, dimensione ed eventuali nomi di riga e di colonna dei dati caricati. (1 punto).

>load("swiss_data.RData")

>ls()

>Swiss

>class(swiss)

>dim(Swiss)

>rownames(Swiss)

>colnames(Swiss)

Domanda 2 (5 punti) . I dati caricati contengono informazioni circa alcuni indicatori socioeconomici di 47 province francofone della Svizzera. Calcolare in due vettori distinti, sotto e sopra, rispettivamente i nomi delle province dove la % di persone cattoliche (colonna Catholic – già riporta la %) è inferiore o uguale alla media, e quelli dove invece tale % è superiore alla media. (3 punti). Usando i due vettori sotto e sopra appena calcolati, tracciare un diagramma a barre della % di persone che hanno superato la scuola primaria (colonna Education – già riporta la %) per le province in sotto e un altro per le province in sopra. Guardate i due diagrammi: si può dire qualcosa circa l'andamento dell'istruzione rispetto all'essere cattolico? (2 punti).

 # CREO VETTORE CHE DA NOMI DI REGIONI CHE HA %catt <= media(catt)

>sotto <- rownames(Swiss)[swiss[,5] <= mean(swiss[,5])]

 # CREO VETTORE CHE HA %catt > media(catt)

>sopra <- rownames(Swiss)[swiss[,5] > mean(swiss[,5])]

 # CREO I DUE DIAGRAMMI

>barplot(swiss[sotto, "Education"], main="diagramma 1", col="lightblue") → swiss[sotto, "Education"] seleziona solo le province del vettore 1 e la colonna Education.

> barplot(swiss[sopra, "Education"], main="diagramma 2", col="red") → swiss[sopra, "Education"] seleziona solo le province del vettore 1 e la colonna Education.

→ posso dire che l'andamento dell'istruzione per le persone sotto la media è maggiore rispetto a sopra la media. Quindi le persone non cattoliche diplomate sono di più rispetto alle cattoliche diplomate.

Domanda 3 (3 punti) . Tracciare un diagramma di dispersione della % di uomini impiegati nell'agricoltura (colonna agricolture – già riporta la %) rispetto alla % di persone che hanno

superato al scuola primaria (colonna Education – già riporta la %). (2 punti). Emerge un andamento/relazione tra i due indicatori? (1 punto).

```
>plot (
        swiss[,2],
        swiss[,4],
        main="diagramma di dispersione",
        col=c("red","blue"), pch=19
        xlab="impiegati agricoltura",
        ylab="hanno superato scuola")
```

→ si nota che gli impiegati all'agricoltura sono maggiori rispetto a quelli che hanno superato la scuola

Domanda 4 (4 punti) . Scrivere un ciclo for che preso in input un vettore numerico vett, di lunghezza n, sommi in una variabile somma gli elementi > 0. (4 punti)

```
>vett <- c(n)
>somma <- 0
>for (i in vett) {
        if(i > 0){
          somma <- somma + i
        }
}
```

APPELLO 12 LUGLIO 2023

Domanda 1 (3 punti). Dopo aver scaricato il file data.csv dal sito upload, caricare i dati in R nella variabile elementi e visualizzarne il contenuto (consiglio: visionare il file prima di caricarlo) (2 punti) Calcolare anche la dimensione dei dati caricati ed eventuali nomi di colonna. (1 punto)

```
>elementi <- read.table("dati.csv", header=T, sep=",")
>elementi
>dim(elementi)
>colnames(elementi)
```

Domanda 2 (3 punti). Calcolare la massa atomica media (colonna atomic.mass) dei gas nobili (valore "noble gas" della colonna metal.or.nonmetal).

>gas_nobili <- elementi[elementi["metal.or.nonmetal."] =="noble gas",] ← prende tutti gli elementi della colonna metal.or.nonmetal. che sono uguali a noble gas, quindi devi selezionare le righe

>media_massa_atomica <- mean(gas_nobili[,"atomic.mass"])

Domanda 3 (4 punti). Contare il numero di elementi presenti in ciascuna categoria distinta della colonna metal.or.nonmetal. (3 punti). Contare il numero di elementi (righe) con massa atomica maggiore di 50. (1 punto)

>categoria <- elementi[,"metal.or.nonmetal."]

>table(categoria) ← conta il numero di elementi per ogni categoria

>n_elementi <- elementi[, "atomic.number"]

>sum(n_elementi > 50)

Domanda 4 (5 punti). Sia dato il codice seguente:

```
f <- function(data, val1=0, val2=1){
        cnt <- -1
        if (is.vector(data)){
                cnt <- 0
        for (val in data){
        if(val >= val1 & val <= val2){
                cnt <- cnt + 1
}}}
return(cnt) }
A <- sample(1:100, 50), f(A, 0), f(A, val1=5), f(A, val2=10), f(A), f(val1=1, val2=2)
```

Dire che operazioni compie la funzione f rispetto ai suoi argomenti, cosa restituisce in output, e se sono presenti errori logici o di sintassi in tutto il frammento di codice (incluse le chiamate ad f). Motivare anche perché secondo voi le chiamate dovessero risultare corrette o meno.

→ la funzione prende in input un data che è un vettore numerico, e val1 e val2 sono numeri di default.

cnt parte da -1, ma se data è un vettore viene impostato che parte da 0.

Se data è un vettore entra in un ciclo for che scorre ogni elemento (val) e chiede se val è compreso tra val1 e val2, se è così cnt aumenta di 1, restituisce poi cnt. Restituisce -1 se data non è un vettore.

In pratica: conta quanti elementi di data sono compresi nell'intervallo [val1, val2].

A <- sample(1:100, 50) → prende 50 numeri casuali dal vettore 1:100

f(A, 0) → conta quanti elementi di A sono compresi tra 0 e 1

f(A, val1=5) → conta quanti elementi sono compresi tra 5 e 1, è una condizione falsa perché val1>val2, da risultato sempre 0

f(A, val2=10) → conta quanti elementi di A sono compresi tra 0 e 10

f(A) → conta quanti elementi di A sono compresi tra 0 e 1

f(val1=1, val2=2) → qui c'è un errore logico in quanto è senza un valore predefinito, perché data non è fornito

APPELLO 20 LUGLIO 2022

Domanda 1 (3 punti). Caricare il dataset state dalla libreria base R e verificare le variabili R che sono state caricate (1 punto). Calcolare classe, dimensione, i nomi di colonna, i nomi di riga dell'oggetto state.x77 (2 punti).

>data(state)

>ls()

>class(state.x77)

>dim(state.x77)

>rownames(state.x77)

>colnames(state.x77)

Domanda 2 (5 punti). Dalla variabile state.x77, che contiene diverse informazioni sugli stati americani, calcolare: la più bassa percentuale perc di analfabetismo (colonna Illiteracy) (2 punti); quali stati hanno proprio la percentuale perc di analfabetismo (colonna Illiteracy). Attenzione: i nomi degli stati americani sono riportati come nomi di riga di state.x77 (3 punti).

>perc <- state.x77[,"Illiteracy"]

>min(perc)

 # TROVO QUALI STATI HANNO QUELLA PERCENTUALE

>stati_perc <- rownames(state.x77)[perc == min(perc)] ← seleziona i nomi delle righe che hanno la percentuale uguale alla min(perc) calcolato prima

>stati_perc

Domanda 3 (5 punti) . Con riferimento alla domanda 3, tracciare l'istogramma dei guadagni medi pro-capite (colonna Income) degli stati che hanno una % di analfabetismo (colonna Illiteracy)

maggiore di 1.5, salvando la figura come file fig1.png (2punti). Ripetere lo stesso esercizio ma per gli stati che hanno la % di analfabetismo minore o uguale a 1.5, salvando la figura come file fig2.png (2 punti). Cosa possiamo dire delle due distribuzioni? L'alfabetismo è correlato al guadagno medio, e se si in che modo? (1 punto)

```
>Income <- state.x77[,"Income"]
>Illiteracy <- state.x77[,"Illiteracy"]
>a <- Income [Illiteracy > 1.5] ← seleziona i numeri delle colonna illiteracy che sono >1.5
>hist(a, xlab="guadagni medi pro-capite", ylab="stati analfabetismo", main="istogramma 1",
col="pink")
>savePlot("fig1.png", type = "png")
>b <- Income [Illiteracy <= 1.5] ← seleziona i numeri delle colonna illiteracy che sono <=1.5
>hist(b, xlab="guadagni medi pro-capite", ylab="stati analfabetismo", main="istogramma 2",
col="lightgreen")
>savePlot("fig2.png", type = "png")
```

Domanda 4 (2 punti). Con riferimento alle domande 3 e 5, si assuma che i valori riportati nella colonna Income siano un campione relativo a un sottoinsieme di stati americani. Si calcoli in R una stima dell'intervallo di confidenza al 95% della media degli Income della popolazione rappresentata da tutti gli stati americani. Si noti che non abbiamo informazioni circa la varianza della popolazione (2 punti).

```
>n <- state.x77[,"Income"]
>media <- mean(a)
>ES <- sd(n) / sqrt(length(n))
>alpha <- 1-0.95
>t <- qt(1 - alpha/2, df = n - 1)
M_E <- t \*ES
>I_C_inf <- media - M_E
>I_C_sup <- media + M_E
>intervallo_confidenza <- c(I_C_sup, I_C_inf)
```

APPELLO 15 GIUGNO 2023

Domanda 1 (3 punti). Caricare dalla libreria base R il dataset cars (1 punto), che contiene l'informazione di velocità al momento della frenata (colonna speed) e lunghezza della frenata stessa (colonna dist) di un insieme di automobili. Visualizzare il contenuto del workspace (1 punto), il tipo e dimensione della/e variabile/i in esso presente/i (1 punto).

```
>data(cars)
>ls()
>dim(cars)
>class(cars)
>str(cars)        ← mostra struttura e tipo di ogni colonna
>cars[,"speed"]
>cars[,"dist"]
```

Domanda 2 (4 punti). Tracciare un diagramma di dispersione delle velocità (colonna speed) rispetto alla lunghezza di frenata (colonna dist) (1 punto).Cosa possiamo dire sulla disposizione dei punti rispetto all'eventuale legame tra l'andamento della velocità e quello della frenata? (2 punti). Tracciare sullo stesso grafico la retta speed=dist (di solito denotata come y=x, o ancora detta bisettrice del primo e terzo quadrante), per aiutarci a interpretare meglio i dati (1 punto).

```
>plot(
        cars[,"speed"],
        cars[,"dist"],
        main="diagramma 1",
        col=c("lightgreen","pink"), pch=19
        xlab="velocità"
        ylab="frenata")
>abline(a=0, b=1, col=1)
```

Domanda 3 (3 punti). Selezionare mediante comandi R La riga corrispondente all'auto con frenata più inefficiente, cioè abbia il rapporto lughezza_frenata / velocità maggiore. Es. Auto A, speed=5, dist=10, rapporto dist/speed = 2 . Auto B, speed=10, dist=100, rapporto dist/speed = 10 . L'auto B è più inefficiente in frenata dell'auto A.

```
>A <- cars[,"speed"]
>B <- cars[,"dist"]
>rapporto <- B/A
```

>max(rapporto) ← verifico quale è il max del rapporto

>which.max(rapporto) ← mi indica la posizione del max nel vettore

>auto_inefficente[which.max(rapporto),] ← mi indica le posizioni dei due numeri usati nel rapporto per il max

Domanda 4 (5 punti). Scrivere una funzione R compare che, presi in input due vettori v1 e v2, restituisca un intero res con le seguenti proprietà:

1. *res vale -1 se la media artitmetica di v1 è minore di quella di v2;*
2. *res vale 0 se la media artitmetica di v1 è uguale a quella di v2;*
3. *res vale 1 se la media artitmetica di v1 è maggiore di quella di v2;*

```
>v1
>v2
>compare <- function(v1,v2) {
        res <-0
        if(mean(v1) < mean(v2)){
                res <- -1
                }
        if (mean(v1) == mean(v2){
                res <- 0
                }
        if (mean(v1) > mean(v2){
                res <- 1
                }
return(res)
}
```

APPELLO 20 GENNAIO 2023

Domanda 1 (4 punti). Caricare in R i dati contenuti nel file sleep_data.tsv (usare l'apposito campo "sep" per dati .tsv – separati da tabulazione) (2 punti). Esso contiene tra l'altro, la variazione di sonno in due gruppi di pazienti (colonna extra), uno di controllo (colonna group = 1) e uno a cui sono stati somministrati dei sonniferi (group = 2). Madiante comandi R calcolare: dimensione dei dati (1 punto); nomi delle colonne (1 punto).

```
>dati <- read.table("sleep.data.tsv", header=T, sep="\t")
```

```
>dim(dati)
>colnames(dati)
```

Domanda 2 (4 punti). In riferimento alla domanda 1, assegnare ai vettori gruppo1 e gruppo2 i valori della colonna extra corrispondenti rispettivamente al gruppo 1 e al gruppo 2 (colonna group) (2 punti). Per i due vettori calcolare: media, primo e terzo quartile (1 punti). Cosa possiamo dire circa gli effetti dell'assunzione del sonnifero rispetto all'incremento di sonno? (1 punto).

```
>gruppo1 <- dati [ ,"extra"] [dati [dati[ , "group"] == "gruppo 1", ]]
>gruppo2 <- dati [ , "extra"] [dati [dati[ , "group"] == "gruppo 2", ]]
>media1 <- mean(gruppo1)
>media2 <- mean(gruppo2)
>q_gruppo1 <- quantile (media1, prob = c(0.25, 0.75))
>q_gruppo2 <- quantile (media2, prob = c(0.25, 0..75))
```

Domanda 3 (4 punti). Creare un vettore numerico vett di 10 elementi, e inizializzare i primi 2 elementi (indici 1 e 2) rispettivamente a 0 e 1 (1 punto). Quindi scrivere un ciclo for che, partendo dalla posizione 3 fino alla fine del vettore, assegni a ciascuna posizione di vett la somma dei due valori in vett precedenti. Il vettore vett alla fine deve contenere i seguenti valori: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34 (3 punti).

```
>vett <- numeric(10)
>vett[1] <- 0
>vett[2] <-1
>for (i in 3:10) {        ← creo il ciclo e lo faccio partire da posizione 3
        vett[i] <-  vett[i-1] + vett[i-2]
} return (vett)
```

Domanda 4 (3 punti). Scrivere una funzione fibonacci che, preso come input un numero intero n (che di default ha valore 10), restituisca un vettore di lunghezza n riempito come descritto nella domanda 6. Per esempio: per n=5, il risultato è un vettore contenente 0, 1, 1, 2, 3; per n=7 il risultato è un vettore contenente 0, 1, 1, 2, 3, 5, 8.

```
>fibonacci <- function (n=10) {
        vett <- numeric(n)
        vett[1] <- 0
```

```
        if (n > 1) {
                vett[2] <- 1 }
        if (n > 2) {
                for (i in 3: n) {
                vett[i] <-  vett[i-1] + vett[i-2]
                }
        } return (vett)
}
```

APPELLO 13 SETTEMBRE 2022

Domanda 1 (3 punti). Caricare il dataset quakes dalla libreria base R e verificare le variabili R che sono state caricate (1 punto). Calcolare classe, dimensione, i nomi di colonna, della variabile quakes (2 punti).

```
>data (quakes)
>ls()
>class (quakes)
>dim (quakes)
>rownames (quakes)
>colnames (quake
```

Domanda 2 (5 punti). Dalla variabile quakes, che contiene informazioni circa i terremoti avvenuti nei pressi delle Isole Fiji, calcolare: il boxplot dell'intensità dei terremoti (colonna mag) avvenuti ad almeno 150 metri, di profondità (colonna depth) (2 punti). Il boxplot dell'intensità dei terremoti (colonna mag) avvenuti a meno di 150 metri di profondità (colonna depth) (2 punti). Cosa possiamo dire circa il legame tra profondità dell'ipocentro e intensità del terremoto? (1 punto).

```
>vett1 <- quakes$mag [quakes [ , "depth"] >= 150]
>vett2 <- quakes$mag [quakes [ , "depth"] < 150]
>boxplot (vett1, main = "Magnitudo terremoti con depth >= 150 km", ylab = "Magnitudo")
>boxplot (vett2, main = "Magnitudo terremoti con depth < 150 km",  ylab = "Magnitudo")
```

Domanda 3 (3 punti). Tracciare un diagramma di dispersione della latitudine (colonna lat) rispetto alla longitudine (colonna long) degli ipocentri dei terremoti (2 punti).
Cosa possiamo dire sulla loro disposizione? (1 punto).

```
>plot (
        quakes[, "lat"],
```

quakes[, “long”],

xlab = "Longitudine",

ylab = "Latitudine",

main = "Ipocentri dei terremoti vicino alle Fiji",

pch = 19, col = "blue"

)

→ Gli ipocentri non sono distribuiti casualmente, ma **si dispongono lungo una fascia allungata**, che corrisponde alla zona di subduzione vicino alle Fiji.

Domanda 7 (4 punti). Dato il seguente frammento di codice:

```
myF <- function(x, conf = 0.95) {
  se <- sd(x) / sqrt(length(x))
  alpha <- 1 - conf
  crit <- qnorm(1-alpha + alpha / 2)
  low <- mean(x) - se * crit
  up <- mean(x) + se * crit
  return(c(low, up))
}
y <- c(-0.5, 0.5, -1.5, 1.2, 0.8, 2, -1.8, -0.3, 0.4, 1.7)
myF(y)
myF(y, conf=0.99)
myF(y^2, conf=0.99)
```

dire che operazione svolge la funzione myF rispetto ai suoi input (3 punti) e se ci sono errori in tutto il frammento di codice (1 punto).

→ la funzione calcola una stima dell’intervallo di confidenza per la media campionaria di x al 95%, assumendo normalità. L’unico errore è la scrittura non precisa del qnorm, che sarebbe qnorm(1-alpha/2).

APPELLO 23 NOVEMBRE 2022

Domanda 1 (2 punti). Caricare il dataset rivers dalla libreria base R e verificare le variabili R che sono state caricate (1 punto). Calcolare classe e lunghezza della variabile rivers (1 punto).

>data(rivers)

>ls()

>class(rivers)

>length(rivers)

Domanda 2 (5 punti). Dalla variabile rivers, che contiene informazioni circa la lunghezza in miglia dei maggiori fiumi nord americani, calcolare: la frazione di dati maggiori della media (2 punti). L'errore standard (1 punto). L'intervallo di confidenza della media usando l'errore standard (2 punti).

```
>media <- mean(rivers)
> dati_maggiori <- sum(rivers > media)
>frazione <- dati_maggiori / length(rivers)
>n <- length (rivers)
>ES <- sd(rivers) / sqrt(n)
>alpha <- 0.05
>crit <- qnorm(1 - alpha/2)   ← quantile normale standard
>I_C_sup <- media + ES*crit
>I_C_inf <- media – ES*crit
>I_C <- c(I_C_sup, I_C_inf)
```

Domanda 3 (3 punti). Possiamo considerare il campione rivers distribuito secondo una distribuzione normale? Mostrare i comandi R necessari per giungere a una risposta, secondo uno dei metodi visti a lezione. (3 punti).

```
>shapiro.test(rivers)
```

Seondo metodo

```
>hist(rivers, main = "Istogramma lunghezza fiumi", xlab = "Miglia")
# QQ-plot
>qqnorm(rivers)
>qqline(rivers, col = "red")
```

Domanda 4 (5 punti). Scrivere una funzione R che preso in input un vettore x e un valore numerico val, restituisca la posizione in x della prima occorrenza di val, se val è presente. In caso contrario, la funzione deve stampare a video il messaggio "val non presente in x". Di default assegnare a val il valore 1.

```
>x <- 1:10
>trova_posizione <- function(x, val=1) {
```

```
        posizione <- which (x == val) [1]
```
← which (x == val) restituisce **gli indici** delle posizioni dove la condizione è TRUE. [1] prende **la prima posizione** trovata, cioè la più a sinistra.

```
        If (val %in% x){
                Return(posizione)
        } else {
                cat ("val non è presente in x \n")
        }
}
```

APPELLO 23 GIUGNO 2022

Domanda 1 (3 punti). Caricare il dataset sleep dalla libreria base R. Esso contiene tra l'altro, la variazione di sonno in due gruppi di pazienti, uno di controllo e uno a cui sono stati somministrati dei sonniferi. Riportare: nome dell'oggetto ivi contenuto (1 punto); tipo e dimensione dell'oggetto (1 punto); nomi delle colonne dello stesso (1 punto).

```
>data(sleep)
>ls()
>sleep

>dim(sleep)
>class(sleep)
>colnames(sleep)
```

Domanda 2 (4 punti). In riferimento alla domanda 1, contare il numero di pazienti del gruppo 1 e quelli del gruppo 2 (1 punto).
Tracciare in un'unica figura il boxplot delle differenze di sonno (colonna extra) per i due gruppi, associando sull'asse orizzontale i due boxplot al nome del rispettivo gruppo (3 punti)

```
>n.pazienti <- sleep[, "group"]
>table(n.pazienti)

>gruppo1 <- sleep[, "extra"] [sleep[, "group"]== 1]
```
← mi seleziona dalla colonna extra solo quelli che nella colonna group sono uguali a 1

```
> gruppo2 <- sleep[, "extra"] [sleep[, "group"]== 2]
```
← mi seleziona dalla colonna extra solo quelli che nella colonna group sono uguali a 2

```
>boxplot (
        gruppo1,
        gruppo2,
        col = c("pink", "yellow"),
        xlab=" "
```
← ho fatto così per togliere i nomi di default perché si sovrapponevano a quelli messi dopo

```
)
```

```
>axis (1, at=c(1,2), labels=c("gruppo1", "gruppo2"))
```

Domanda 3 (4 punti). Scrivere una funzione minmax che preso in input un vettore v, restituisca un vettore res che ne contiene il minimo e il massimo. La funzione deve controllare che v contenga almeno 2 elementi, in caso contrario non deve fare nulla (3 punti). Chiamare poi minmax sulle differenze di sonno del gruppo 2 (rif. domande 1 e 3) (1 punto).

```
>min_max <- function(v){
        if (length(v) < 2) {
                return (NULL)
        }
        min <- min(v)
        max <- max(v)
        res <- c(min, max)
return (res)
}

>gruppo2 <- sleep[, “extra”] [sleep[, “group”]== 2]        ← seleziono da colonna extra gruppo 2
>min_max (gruppo2)
```

Domanda 4 (4 punti). Assumendo che il vettore x contenga le misurazioni del numero di visitatori del museo d’arte nell’ultimo anno, contare (in una variabile cnt) il numero di giorni in cui tale numero è la metà (o meno) rispetto a quello del giorno precedente (4 punti).

```
>n <- length(x)
>cnt <- 0        ← inizializziamo un contatore
>for (i in 2:n) {        ← inizia il ciclo dal secondo giorno fino all’ultimo
        if (x[i] <= x[i-1] / 2) {        ← verifico se il numero di visitatori è metà o meno rispetto al giorno precedente
                cnt <- cnt +1        ← se la condizione è vera incrementa il contatore di uno
        }
}
>cnt
```

APPELLO 14 FEBBRAIO 2025

Domanda 1 (3 punti). Dopo aver scaricato il file dati.csv dal sito upload, caricare i dati in R nella variabile dati e visualizzarne il contenuto (consiglio: visionare il file prima di caricarlo).

(2 punti) Calcolare anche la dimensione dei dati caricati ed eventuali nomi di riga e di colonna. (1 punto)

```
>dati <- read.table ("dati.csv", header=T, sep=",")
>ls()
>dati
>dim(dati)
>rownames(dati)
>colnames(dati)
```

Domanda 2 (3 punti) . Tramite comandi R, calcolare il numero di totale di metalli alcalini presenti (colonna metal.or.nonmetal, valore alkali metal).

```
>metalli_alcalini <- dati [, “metal.or.nonmetal.”] == “alkali metal”
>sum (metalli_alcalini)
```

Domanda 3 (5 punti). Tramite comandi R, calcolare la massa atomica media (colonna atomic.mass) di ciascuna categoria di elementi (valori distinti nella colonna metal.or.nonmetal) (4 punti), inserendo le corrispondenti medie in un vettore medie di lunghezza esattamente pari al numero di categorie di elementi distinte, impostandone l’attributo names proprio a tali categorie. Chiedere al docente se non fosse chiara la richiesta. (1 punto)

```
>categorie <- table(dati[, “metal.or.nonmetal.”]

>categoria1 <- dati[dati["metal.or.nonmetal."] =="actinoid", ]      ← seleziono solo i dati che hanno nella colonna metal.or.nonmetal. actinoid, devo selezionare le righe
>atomic_mass1 <- categoria1[, “atomic.mass”]      ← seleziono solo colonna atomic.mass
>media1 <- mean(atomic_mass1)                     ← calcolo la media

>categoria2 <- dati[dati["metal.or.nonmetal."] =="halogen", ]
>atomic_mass2 <- categoria2[, “atomic.mass”]
>media2 <- mean(atomic_mass2)

>categoria3 <- dati[dati["metal.or.nonmetal."] =="metalloid", ]
>atomic_mass3 <- categoria3[, “atomic.mass”]
>media3 <- mean(atomic_mass3)

>categoria4 <- dati[dati["metal.or.nonmetal."] =="transition metal", ]
>atomic_mass4 <- categoria4[, “atomic.mass”]
>media4 <- mean(atomic_mass4)

>categoria5 <- dati[dati["metal.or.nonmetal."] =="alkali metal", ]
>atomic_mass5 <- categoria5[, “atomic.mass”]
>media5 <- mean(atomic_mass5)

>categoria6 <- dati[dati["metal.or.nonmetal."] =="lanthanoid", ]
```

```
>atomic_mass6 <- categoria6[, “atomic.mass”]
>media6 <- mean(atomic_mass6)

>categoria7 <- dati[dati["metal.or.nonmetal."] =="noble gas", ]
>atomic_mass7 <- categoria7[, “atomic.mass”]
>media7 <- mean(atomic_mass7)

>categoria8 <- dati[dati["metal.or.nonmetal."] =="alkaline earth metal", ]
>atomic_mass8 <- categoria8[, “atomic.mass”]
>media8 <- mean(atomic_mass8)

>categoria9 <- dati[dati["metal.or.nonmetal."] =="metal", ]
>atomic_mass9 <- categoria9[, “atomic.mass”]
>media9 <- mean(atomic_mass9)

>categoria10 <- dati[dati["metal.or.nonmetal."] =="nonmetal", ]
>atomic_mass10 <- categoria1[, “atomic.mass”]
>media10 <- mean(atomic_mass10)

>medie <- c(media1, media2, media3, media4, media5, media6, media7, media8, media9, media10)
>names(medie) <- c(“acnoid”, “halogen”, “metalloid”, "transition metal", "alkali metal",
“lanthanoid”, “noble gas”, “alkaline earth metal”, “metal”, “nonmetal” )
```

Domanda 4 (4 punti). Scrivere una funzione R freq_basi, che presa in input una stringa DNA sull’alfabeto {A,T, G, C} (si può assumere che ciò sia vero, si può non controllarlo), conti le occorrenze di ciascuno dei simboli dell’alfabeto. Il risultato va inserito in un vettore freq, che riserva una posizione per ogni simbolo dell’alfabeto.
Per esempio, se DNA contenesse la stringa ATTGCGGATTAAA, allora il vettore freq conterrà i valori 5, 4, 3, 1, rispettivamente numero di occorrenze del simbolo A, T, G e C.

```
>freq_basi <- function (DNA) {
        freq <- c(A=0, T=0, G=0, C=0)
        n <- nchar(DNA)        ← mi conta lunghezza stringa
        for (i in 1:n){
                b <- substring(DNA, i, i)        ← estrae una sottostringa da DNA che prende tutti i singoli caratteri separatamente
                if (b == “A”){
                        freq[“A”] <- freq[“A”] + 1
                }else if (b == “T”){
                        freq[“T”] <- freq[“T”] + 1
                }else if (b == “C”){
                        freq[“C”] <- freq[“C”] + 1
```

```
        }else if (b == "G"){
            freq["G"] <- freq["G"] + 1
        }
    }
return(as.vector(freq))        ← restituisce i conteggi come vettore numerico
}
>DNA <- c("AATTCCGG")
>freq_basi(DNA)
```

APPELLO 17 GENNAIO 2024

Domanda 1 (3 punti). Caricare il dataset pressure dalla libreria standard di R (1 punto), che contiene informazioni circa la temperatura (colonna temperature) e la pressione (colonna pressure) del vapore di mercurio. Visualizzare il contenuto del workspace (1 punto), il tipo e dimensione delle variabili in esso presenti (1 punto).

```
>data(pressure)
>ls()
>class(pressure)
>mode(pressure)
>dim(pressure)
```

Domanda 2 (3 punti) . Calcolare nel vettore ratio il rapporto temperatura/pressione, e determinarne la posizione del valore minimo e del valore massimo. (2 punti)
Usare tali posizioni per accedere agli elementi corrispondenti dei vettori pressione e temperatura (1 punto).

```
>ratio <- pressure[, 1] / pressure[, 2]
>massimo <- max (ratio)
>minimo <- min(ratio)
>which (ratio == massimo)        ← mi dice la posizione nel vettore ratio del valore max
>which(ratio == minimo)          ← mi dice la posizione nel vettore ratio del valore min

> pressure[which.max(ratio), ]
>pressure[which.min(ratio), ]
```

Domanda 3 (4 punti). Calcolare la temperatura media nella variabile avgtemp, quindi calcolare i valori medi di pressione quando la temperatura è minore di avgtemp, e quando è maggiore o uguale ad avgtemp, rispettivamente (3 punti).
Cosa possiamo dire sul comportamento reciproco di pressione e temperatura dai valori riscontrati? (1 punto)

```
>pressione <- pressure [, 2]
>temp <- pressure [, 1]
>avgtemp <- mean(temp)

>i <- pressione < avgtemp          ← ho trovato i numeri della pressione che sono < della media
> pressione[i]
>mean(i)        ← ho trovato la media della pressione bassa
>b <- pressione >= avgtemp         ← ho trovato i numeri della pressione che sono >= della media
> pressione[b]
>mean(b)        ← ho trovato la media della pressione alta
```

→ possiamo dire che la pressione bassa > pressione alta e quindi significa che la pressione diminuisce con l'aumentare della temperatura. Quando invece la pressione la pressione alta > pressione bassa, la pressione aumenta all'aumentare della temperatura.

Domanda 4 (5 punti). Sia dato il codice seguente:

```
f <- function(x, target=1){
        cum <- 0
        res <- 0
        for (i in 1:length(x)){
                cum <- cum + x[i]
                if (cum >= target && res == 0) {
                        res <- i
                }
        }
return(res)
}
A <- sample(1:100, 20)
f(A, 1000)
f(x=A)
f(target=1)
```

Dire che operazioni compie la funzione f rispetto ai suoi argomenti, cosa restituisce in output, e se sono presenti errori logici o di sintassi in tutto il frammento di codice (incluse le chiamate ad f). Motivare anche perché secondo voi le chiamate sono corrette o meno. (5 punti)

→ la funzione inizializza due variabili: cum e res a 0. Scorre ogni elemento di x con un ciclo for e ogni volta aggiunge a cum l'elemento corrente (l'elemento che il ciclo sta analizzando). Appena cum raggiunge o supera il valore di target per la prima volta salva l'ultimo indice analizzato che è arrivato o a superato il valore di target i in res. Alla fine, restituisce res.

La funzione quindi restituisce l'indice del primo elemento del vettore x per cui la somma cumulativa raggiunge o supera target. Se la somma cumulativa non raggiunge mai target, restituisce 0.

A <- sample(1:100, 20) → genera 20 numeri casuali da 1 a 100, non ci sono errori

LE CHIAMATE:
f(A, 1000) → prende i valori di A e imposta come target 1000, nessun errore

f(x=A) → espliciti che x nella funzione prende il valore di A e target ha come default 1, non ci sono errori
f(target=1) → qui c'è un problema, x non viene imposto, e la funzione richiede un x in input, R da quindi un errore di esecuzione

APPELLO 24 GENNAIO 2025

Domanda 1 (3 punti). Scaricare il file dati.RData e caricarne il contenuto in memoria mediante apposito comando. (1.5 punti) Visualizzare cosa è stato caricato, la rispettiva classe e dimensione. (1.5 punti)

```
>load("dati.RData")
>ls()
>class(v1)
>length(v1)    ← uso length perché sono vettori e non matrici
>class(v2)
>length(v2)
>class(v3)
>length(v3)
```

Domanda 2 (4 punti) . Applicare un test statistico per verificare che ciascuno dei campioni caricati sia distribuito o meno secondo una distribuzione normale. Riportare i comandi utilizzati e l'interpretazione del risultato che ha condotto a ritenere il campione distribuito normalmente o meno.

```
>shapiro.test(v1)

>shapiro.test(v2)

>shapiro.test(v3)
```

→ Se il test di normalità non è significativo, quindi ha un p-value > 0,05, accettiamo l'ipotesi nulla ovvero i dati sono distribuiti normalmente. Solo nel caso in cui il test è statisticamente significativo, quindi p-value < 0,05, possiamo rifiutare la normalità.
Da tutti e tre i test notiamo che i p-value sono tutti > 0.05, quindi tutti e tre i vettori hanno una distribuzione normale.

Domanda 3 (4 punti). Scrivere un ciclo for che dati due vettori vett1 e vett2, verifichi che il secondo vettore cominci esattamente con gli elementi contenuti in vett1. Ad esempio, se avessimo vett 1: 0, 2, 5, -1, 3 e vett 2: 0, 2, 5, -1, 3, 12, 1 5, alla fine del ciclo la risposta deve essere TRUE. Se invece per esempio avessimo vett1: "A", "C", "T" e vett 2 : "C", "A", "T", "T", "G " , alla fine del ciclo la risposta dove essere FALSE. Per semplicità, si assuma che vett1 e vett2 esistano già.

```
>vett1 <- c(…)
>vett2 <- c(…)
>inizio_uguale <- TRUE      ← assumo che l'inizio di vett2 è uguale a quello di vett1
```

```
>for (i in 1:length(vett1)) {   ← analizza tutti gli indici/elementi di vett1
        if(vett1[i] != vett2[i]) {      ← se l'indice di vett1 è diverso dall'indice di vett2
        inizio_uguale <- FALSE
        break                   ← esce dal ciclo
  }
}
```

Domanda 4 (5 punti) . Sia dato il codice seguente:

```
f <- function(input){
        res <- FALSE
        if (is.character(input)){
                n <- nchar(input)
                cnt = 0
                for (i in 1:floor(n/2)){
                        if (substr(input, i, i) == substr(input, n-i+1, n-i+1)){
                                cnt <- cnt + 1
                        }
                }
                if (cnt == floor(n/2)){
                        res <- TRUE
                }
        }
return(res)
}
v <- "CASASAC"
f(v)
f(input = v)
v <- "ESAME"
f(v)
```

Dire che operazioni compie la funzione f rispetto al suo argomento, cosa restituisce in output, e se sono presenti errori logici o di sintassi in tutto il frammento di codice (incluse le chiamate ad f). Motivare anche perché le chiamate risultano corrette o meno.

→ la funzione assume il vettore res come FALSE (il risultato di default è FALSE). Se i valori del vettore input sono dei caratteri mi da come risposta TRUE e va avanti a eseguire, Se non è una stringa, restituisce direttamente FALSE, calcola poi la lunghezza della stringa.

Successivamente fa un ciclo for confronta i caratteri della stringa in posizione simmetrica, ovvero il primo con ultimo, secondo con penultimo, ecc. Se i caratteri corrispondono, incrementa il contatore cnt. Se tutte le coppie coincidono (cnt == floor(n/2)), allora res diventa TRUE.

Quindi la funzione f serve per verificare se una stringa è palindroma (uguale letta da sinistra e da destra).

```
v <- "CASASAC"
f(v)
                ← ha creato la stringa e ha richiamato la funzione, non ci sono errori
f(input = v)    ← specifica esplicitamente il nome dell'argomento (input).
```

v <- “ESAME”

←crea la stringa e richiama la funzione, non ci sono errori

f(v)

Errore presente: uso di virgolette tipografiche (“ ”) al posto di quelle corrette (" ").

APPELLO 27 NOVEMBRE 2024

Domanda 1 (3 punti). Scaricare il file Data.csv e caricarne il contenuto in memoria mediante apposito comando. (1.5 punti)
Calcolare anche classe, dimensione ed eventuali nomi di riga e di colonna dei dati caricati. (1.5 punti)

```
>dati <- read.table (“Data.csv", header=T, sep=",")
>class(dati)
>dim(dati)
>rownames(dati)
>colnames(dati)
```

Domanda 5 (4 punti) . Calcolare l’indice del rinoceronte che ha il rapporto peso/lunghezza maggiore, e l’indice di quello che ha il rapporto peso/lunghezza minore. (1.5 punti). Selezionare tutti i rinoceronti che hanno, contemporaneamente, un peso sopra la media e una larghezza del collo sopra la media. (1.5 punti). Tracciare, nella stessa figura, un diagramma di dispersione della lunghezza rispetto al peso, prima per tutti i rinoceronti, e poi solo per i rinoceronti selezionati al punto precedente. (1 punto).

```
>rapporto <- dati[, 2] / dati[, 3]
>massimo <- max (rapporto)
>minimo <- min(rapporto)
>peso_media <- mean(dati[,3])
>larghezza_collo_media <- mean(dati[,1])
>rinoceronti_selezionati <- which(peso_media < dati[,3] & larghezza_collo_media < dati[,1] )

>par(mfrow = c(2,1))
>plot (
        dati[,2], dati[,3],
        pch=19,
        col = c(“pink”, “green”)
)
>plot (
        rinoceronti_selezionati,
        pch=20;
        col=”blue”
)
```

Domanda 3 (4 punti) . Scrivere un ciclo for che dato un vettore numerico vett, calcoli tutti gli indici j del vettore dove vett è compreso tra 0 e 5 (compresi). Se, ad esempio, il vettore

fosse vett: 2, -5 , 2.4, 3.5, 10 , 1 , il ciclo dovrebbe calcolare il vettore contenente i valori 1, 3, 4, e 6. Per semplicità, si assuma che vett esista già. (4 punti)

```
>vett <- c(1,3,5,2,7,3,8)
>indici <- c()
>for (j in 1:length(vett)){
        if(vett[j] >=0 & vett[j] <= 5){        ← controllo sel'elemento in posizione j è tra 0 e 5
                indici <- c(indici, j)         ← aggiunge l'indice j al vettore indici
        }
}
>indici          ← mi da un vettore con la posizione dei numeri che sono compresi tra 0 e 5
```

Domanda 4 (4 punti). Sia dato il codice seguente:

```
f <- function(A, v1=0, v2){
        res <- 0
        if (is.numeric(A)){
                n <- length(A)
                if (n > 0){
                        for (i in 1:n){
                                for (j in (i+1):n){
                                        if(A[i] == 2*A[j] || A[i] == v1 + v2){
                                        res <- res + 1
                                        }
                                }
                        }
                }
        }
return(res)
}
v <- seq(1,100, by=2)
f(v)
f(v, v1=1)
f(v, 1, 5)
f(v, v2=5)
```

Dire che operazioni compie la funzione f rispetto ai suoi argomenti, cosa restituisce in output, e se sono presenti errori logici o di sintassi in tutto il frammento di codice (incluse le chiamate ad f). Motivare anche perché le chiamate risultano corrette o meno.

→ la funzione prende in input 3 argomenti.

Se i valori di A sono numerici verifica se la lunghezza del vettore A sia maggiore di 0. Se è così entra in un ciclo for che analizza tutti i dati da uno a lunghezza di A, successivamente entra in un secondo ciclo for che conta gli elementi dalla seconda posizione (i+1) alla lunghezza del vettore A.

Scorre quindi tutte le coppie (i, j). Se l'indice di A è uguale a 2 per l'indice di A oppure è uguale a v1+v2 (sono gli altri due elementi presi in input, v1 ha valore di defaul 0), allora res incrementa di 1.

Quindi la funzione conta quante coppie (i,j) rispettano almeno una delle due condizioni.

v <- seq(1,100, by=2)
f(v) ← questa chiamate è un errore perché v2 non ha un valore di default e quindi la chiamata genera un errore

f(v, v1=1) ← anche qui c'è lo stesso errore, manca v2

le altre chiamate sono corrette perché presentano tutti e tre gli argomenti un valore.

Errore pratico ← in f la funzione è problematica perché in R, tutti gli argomenti **senza default** diventano obbligatori, quindi chiamate come `f(v)` o `f(v, v1=1)` falliscono. Per funzionare, bisognerebbe dare un default anche a `v2` (es. `v2=0`).

APPELLO 16 NOVEMBRE 2023

Domanda 1 (3 punti). Scaricare il file misurazioni.txt dal sito upload, e caricare nella variabile R dati i dati in esso contenuti. (2 punti)
Calcolare lunghezza, media, valore minimo e massimo della variabile dati. (1 punto)

```
>dati <- scan("misurazioni.txt", what=numeric())
>length(dati)
>mean(dati)
>min(dati)
>max(dati)
```

Domanda 2 (3 punti) . Tracciare un histogramma di dati (1 punto) e dire il numero di utenti connessi in quale intervallo cade più di frequente. (2 punti)

```
>hist (dati, xlab="utenti", ylab="frequenza", col="pink", cex.axi=1.5, cex.lab=1.5)
> h <- hist(dati, plot=FALSE)          ← Istogramma senza disegno, solo per i conteggi
> intervallo <- h$mids[which.max(h$counts)]          ← Trovo l'intervallo con frequenza massima
```

Domanda 3 (4 punti) . Creare un vettore diff_puntuale della stessa lunghezza di dati, contenente in ogni posizione i (eccetto la prima) la differenza dell'elemento i-mo di dati con quello immediatamente precedente (sempre in dati). Utilizzare un ciclo for per assegnare le posizioni di diff_puntuale. Il primo elemento di diff_puntuale coincide invece col primo elemento di dati.

```
>diff_puntuale <- numeric(length(dati))        ← ho creato il vettore vuoto di lunghezza come dati
>diff_puntuale[1] <- dati[1]                   ← ho impostato il primo elemento = a quello di dati
>for (i in 2:length(dati)){
        diff_puntuale[i] <- dati[i] – dati[i-1]
}
>diff_puntuale
```

Domanda 4 (5 punti) . Sia dato il codice seguente:

```
f <- function(data1, data2, opz = 1){
        res = NA
        if (is.vector(data1) && is.vector(data2)){
                if(length(data1) == length(data2)){
                        if(opz == 1){
                        res <- data1 + data2
                        }else{
                        res <- data1 - data2
                        }
                }
        }
return(res)
}
A <- sample(1:100, 50)
B <- sample(1:100, 50)
f(A, B)
f(A, B, 0)
f(A, 1)
```

Dire che operazioni compie la funzione f rispetto ai suoi argomenti, cosa restituisce in output, e se sono presenti errori logici o di sintassi in tutto il frammento di codice (incluse le chiamate ad f). Motivare anche perché secondo voi le chiamate sono corrette o meno. (5 punti)

→ Inizializza res a NA, ovvero non ha nessun elemento. Controlla se entrambi gli argomenti sono vettori (is.vector); Se hanno la stessa lunghezza: se opz == 1 → fa la somma elemento per elemento altrimenti → fa la differenza elemento per elemento. Restituisce res.

f(A, B)
f(A, B, 0)
Entrambe le chiamate sono corrette, nella prima opz non è passato, quindi resta al valore predefinito 1 e il risultato è somma per elemento, nel secondo invece opz è 0 e qundi va nell'else.
f(A, 1)
Questa chiamata non ha errori di sintassi, ma le lunghezze dei vettori sono diverse, e quindi restituisce NA, non fa nessuna operazione utile.

APPELLO 9 SETTEMBRE 2025

Domanda 1 (3 punti). Caricare in memoria il dataset airquality dalla libreria base di R. (1 punto). Calcolare la dimensione del dataset e verificare la presenza di nomi di riga e colonna. Inoltre, stampare i primi 6 valori dell'attributo Ozone. (2 punti)

>data(airquality)

>dim(airquality)

>rownames(airquality)

>colnames(airquality)

>airquality[1:6,"Ozone"] → stampa i primi 6 valori della colonna "Ozone"

Domanda 2 (5 punti). Il dataset contiene, tra le altre informazioni, i valori del vento (colonna Wind, miglia orarie), della temperatura (colonna Temp, gradi Fahrenheit), il mese (Month) e il giorno del mese (Day). Calcolare le medie mensili della temperatura e del vento. Quindi avremo un valore medio di temperatura e uno di vento per ogni mese presente nella colonna Month*. (2 punti) Trasformare le medie delle temperature da gradi Fahrenheit (F) in gradi Celsius (C), ricordando la formula di conversione: C = (F - 32) * (5/9) (1 punto). Rappresentare le medie mensili di temperatura Celsius e di vento mediante un grafico a barre raggruppate. (1 punto)*
Cosa emerge dall'andamento delle due grandezze? Esiste un legame tra di esse? (1 punto)

>medie_temp <- tapply (airquality[,"Temp"], airquality[,"Month"], mean)

>medie_wind <- tapply (airquality[,"Wind"], airquality[,"Month"], mean)

→calcolo con tapply() le medie dei vari mesi del tempo e del vento

>medieTemp_C <- (medie_temp - 32) * (5/9) ←converto in celsius

>par(mfrow = c(1,2))

>barplot(medieTemp_C, main="Temp in celsius", ylab="celsius", xlab="mesi", col=pink)

>barplot(medie_wind, main="vento", ylab="vento", xlab="mesi", col=blue)

→dall'andamento delle due grandezze emerge il fatto che nel mese 5 la temperatura è più bassa rispetto al vento, mentre negli altri mesi la temperatura aumenta rispetto al vento, e poi al mese 9 diminuisce ancora. Vuol dire quindi che la temperatura sale nei mesi più caldi e scende in quelli più freddi.

Si nota inoltre che il vento in confronto alla temperatura tende a diminuire quando essa aumenta, e ad aumentare quando la temperatura diminuisce, quindi il vento è inversamente proporzionale alla temperatura.

Domanda 3 (3 punti). Calcolare, se ne esistono, i giorni (righe) in cui sia le temperature che il vento sono sopra le rispettive medie. Il risultato deve riportare per tali giorni il mese e il giorno del mese corrispondente.

>media_temp <- mean(airquality[,"Temp"])

>media_wind <- mean(airquality[,"Wind"])

> sopra_media <- subset(airquality, Temp > media_temp & Wind > media_wind) →subset() filtra le righe che rispettano una condizione logica

>risultato <- sopra_media[c("Month", "Day")]

Domanda 4 (4 punti). Scrivere in linguaggio R una funzione conta_occorrenze, che presa in input una stringa string, e una seconda stringa cerca, restituisca quante volte cerca compare come sottostringa di string. Esempio: string: ATTGATTGCACATTGCCTG cerca: TTG. La funzione deve restituire il valore 3, corrispondente alle occorrenze in grassetto.

```
>conta_occorrenze <- function(sting, cerca) {
        n <- nchar(string)      ←conta le lettere nella riga string
        m <- nchar(cerca)       ←conta le lettere nella riga cerca
        contatore <- 0
        for ( i in 1:(n-m+1)) {
                pezzo <- substr(string, i, i+m-1)      ←prende un pezzetto lungo come la parola da cercare
                if (pezzo == cerca) {
                        contatore <- contatore + 1
                }
        }
        return (contatore)
}
```