UNIVERSITÀ DEGLI STUDI DELLA TUSCIA

# PROGRAMMAZIONE

## Linguaggio R
## Strutture selettive e iterative

*Dott. Franco Liberati*
*franco.liberati@unitus.it*

# LINGUAGGIO R

## Argomenti del corso

# LINGUAGGIO R

## Strutture: condizione e iterazione

- Un algoritmo è eseguito rispettando l'ordine in cui le istruzioni sono scritte dal programmatore, partendo dalla prima istruzione fino a raggiunge l'ultima
- Alcune volte, però, in accordo al verificarsi o meno di una certa condizione, l'istruzione da eseguire **può non essere quella sequenziale**. Per esempio un algoritmo può effettuare un controllo prima di una divisione per escludere il caso in cui il divisore sia uguale a zero.
- Per consentire di risolvere queste situazioni bisogna sfruttare delle **strutture di controllo condizionali**, cioè delle istruzioni che permettono di incanalare il flusso del programma in relazione alla veridicità di una affermazione
- La maggior parte degli algoritmi, inoltre, richiede l'esecuzione ripetuta di sequenze di istruzioni, cioè si effettua un ciclo (o una iterazione), fino a quando una condizione di continuazione rimane vera

# LINGUAGGIO R

## Strutture: condizione e iterazione

- Esistono due modalità di iterazione: **controllata da un contatore** (o iterazione definita), nella quale è noto in anticipo quante volte il ciclo è eseguito (FOR), e **validata da sentinella** (o iterazione indefinita), nella quale non si può determinare a priori il numero di ripetizioni del ciclo (WHILE, da cui si deriva anche il controllo condizionale IF)
- I sistemi di iterazione, oltre ad essere essenziali per la risoluzione degli algoritmi, offrono al programmatore una gestione semplice e immediata, come la lettura e la manipolazione di strutture dati sequenziali, quali i vettori e le stringhe, o quelle dinamiche, come le liste

# LINGUAGGIO R

## Controllo condizionale IF THEN ELSE

- Il **controllo condizionale** è una struttura di selezione che esegue un trasferimento del controllo del codice se una condizione è vera; oppure procede in un altro blocco di istruzioni, se la condizione è falsa.
- La sintassi in R è:

```
if (condizione) {ISTRUZIONI
} else {ISTRUZIONI}
```

*Se la condizione risulta vera, fa eseguire l'istruzione, o il blocco di istruzioni, specificata tra parentesi graffe; altrimenti (se la condizione è falsa) si procede nell'esecuzione delle istruzioni presenti nel ramo else (qualora il ramo else mancasse si proseguirebbe con le istruzioni successive appartenenti al programma)*

*Nel caso in cui la condizione else non sia verificata l'esecuzione del programma continua nella modalità sequenziale*

# LINGUAGGIO R

## Controllo condizionale IF THEN ELSE: esempio

Stampa di un messaggio nel caso di due operandi uguali

```
#Inizializzazione primo oggetto (vettore atomico intero)
valore1 <- 20L
#Inizializzazione secondo oggetto (vettore atomico intero)
valore2 <- 20L
#Confronto di uguaglianza e scrittura
if (valore1==valore2) {
                        cat("valore1 (",valore1,") è uguale a valore2 (",valore2,")\n")
                        }

#Esecuzione dello script e risultato
valore1 (20) è uguale a valore2 (20)
```

# LINGUAGGIO R

## Controllo condizionale IF THEN ELSE: esempio

### Massimo tra due operandi

```
#Inizializzazione primo oggetto
valore1 <- 10L
#Inizializzazione secondo oggetto
valore2 <- 20L
#Confronto di uguaglianza e impostazione del massimo
if (valore1>valore2) {
                max<-valore1
                } else {max<-valore2}
#Visualizzazione del massimo
print(max)
#Esecuzione dello script e risultato
[1] 20
```

VARIANTE PROPOSTA IN AULA DA UNO STUDENTE

```
#Inizializzazione primo oggetto
valore1 <- 10L
#Inizializzazione secondo oggetto
valore2 <- 20L
#Confronto di uguaglianza e impostazione del massimo
if (valore1==valore2) {cat("I VALORI SONO UGUALI\n");
}else {if (valore1>valore2) {
                max<-valore1
                } else {max<-valore2}
    }
#Visualizzazione del massimo
print(max)
#Esecuzione dello script e risultato
[1] 20
```

# LINGUAGGIO R

## Controllo condizionale IF THEN ELSE (variante)

- Il linguaggio R offre anche la funzione **ifelse (*test*,*yes*,*no*)**; dove ***test*** indica la condizione e ***yes*** e ***no*** i due blocchi di istruzioni rispettivamente nel caso di condizione vera oppure falsa

**ifelse** (condizione, yes,no)

### Massimo tra due operandi

```
valore1 <- 10L
valore2 <- 20L
#Confronto di uguaglianza e impostazione del massimo
ifelse (valore1>valore2,max<-valore1,max<-valore2)
#Visualizzazione del massimo
print(max)
#Esecuzione dello script e risultato
[1] 20
```

# LINGUAGGIO R

## Controllo condizionale IF THEN ELSE (variante)

### Massimo tra due operandi

```
valore1 <- 10L
valore2 <- 20L

#Confronto di uguaglianza e impostazione del massimo
ifelse (valore1>valore2,max<-valore1,max<-valore2)

#Visualizzazione del massimo
print(max)

#Esecuzione dello script e risultato
[1] 20
```

# LINGUAGGIO R

## Controllo condizionale IF THEN ELSE: annidamento

- La struttura di selezione può essere usate contiguamente creando quello che si chiama **annidamento**
- Non c'è uno schema definito perché l'annidamento è variabile ed è caratteristico dell'algoritmo in cui si concretizza. Un caso di annidamento minimo può essere così descritto:

```
if (condizione) {
        if (condizione) {
                istruzioni
        } else {istruzioni}
} else {istruzioni}
```

# LINGUAGGIO R

## Controllo condizionale IF THEN ELSE annidamento: esempio

### Massimo tra tre operandi

```
#Inizializzazione oggetti
valore1 <- 10L; valore2 <- 30L; valore3 <- 20L
#Confronti
if (valore1>valore2) {
        if (valore1>valore3) {max<-valore1}
        else{max<-valore3}
                }
        else {
                if (valore2>valore3) {max<-valore2}
                else{max<-valore3}
        }
print(max)
#Esecuzione dello script e risultato
[1] 30
```

# LINGUAGGIO R

## Controllo condizionale IF THEN ELSE annidamento: esempio 2

### Massimo tra tre operandi

```
#Inizializzazione oggetti
valore1 <- 10L; valore2 <- 30L; valore3 <- 20L;
#Creazione di un oggetto che esegue il secondo confronto
cond1<-ifelse(valore1>valore3,max<-valore1,max<-valore3)
#Creazione di un oggetto che esegue il terzo confronto
cond2<-ifelse(valore2>valore3,max<-valore2,max<-valore3)
#Primo confronto e attivazione dell'oggetto in relazione alla veridicità della condizione
ifelse(valore1>valore2,cond1,cond2)
print(max)
#Esecuzione dello script e risultato
[1] 30
```

Il programma mostra la potenza del concetto di oggetto che non solo assume valori numerici e strutture dati, ma svolge anche il ruolo di contenitore di funzione.

# LINGUAGGIO R

## Ripetizione: WHILE

- Un linguaggio di programmazione ha un costrutto che realizza una **struttura iterativa condizionata** che esegue lo stesso blocco di codice fin quando si verifica un particolare evento
- La sintassi è:

```
while (condizione)
        {
        ISTRUZIONI
        }
```

*Il blocco delle istruzioni si ripete fino a quando la condizione da TRUE. Non appena la condizione restituisce FALSE il programma non considera più il blocco di istruzioni e prosegue il suo andamento sequenziale*

# LINGUAGGIO R

## Ripetizione: WHILE esempio

Stampa di numeri pari da 100 a 0

```
#inizializzazione dell'oggetto
n<-100

#Si ripetono le istruzioni nel corpo del while
#fino a quando il valore (numerico) è
#maggiore o uguale a zero
while(n>=0){
        cat(n,"\n") #stampa del valore
corrente
        n<-n-2   #decremento
}
```

```
#VARIANTE
#inizializzazione dell'oggetto
n<-101

while(n>=0){
        if (n%%2==0){cat(n, "\n");}
        n=n-1;
}
```

# LINGUAGGIO R

## Ripetizione: WHILE esempio

Calcolo del fattoriale, ovvero una funzione, rappresentata con il punto esclamativo, che opera moltiplicando tutti i numeri interi e positivi tra il numero esplicitato come argomento e il valore 1; cioè n!=n·n-1·n-2·…·3·2·1

```
#Lettura valore da tastiera (con messaggio informativo per l'utente)
operando<-readline(prompt="Inserire un valore per calcolo fattoriale: ")
operando<-as.integer(operando)
#Inizializzazione
fattoriale<-1
#Calcolo del fattoriale mediante ripetizione indefinita
while (operando>1){
    fattoriale<-fattoriale*operando
    operando<-operando-1
}
print(fattoriale)
#Stampa risultato con lettura valore da tastiera 6
[1] 720
```

# LINGUAGGIO R

## Ripetizione: WHILE

- L'impiego di questo costrutto può comportare un **ciclo infinito**, ovvero la condizione di continuazione non diventa mai falsa e quindi il programma ripete incessantemente il blocco delle istruzioni
- Un ciclo infinito è sempre una condizione da scongiurare perché, oltre ad essere un'operazione inutile, comporta un drastico rallentamento dell'elaboratore che perde cicli macchina per svolgere sempre le stesse, e vane, istruzioni

**ESEMPIO**

```
indice=0L;
pari=0L;
while (indice<10){
                cat("\n",pari);
                pari=pari+2;
}
```

---

```
0
2
3
4
6
8
10
12
14
16
18
20
...
NON FINSICE MAI!!!!!
```

# LINGUAGGIO R

## Ripetizione forzata: REPEAT… IF

- La struttura di iterazione detta **ripetizione forzata** verifica la condizione di continuazione dopo l'esecuzione del corpo (interno alla ripetizione forzata)
- Pertanto, il blocco iterativo è sempre eseguito almeno una volta; e quando la condizione non è verificata l'esecuzione prosegue normalmente
- Nel Linguaggio R una ripetizione ha sintassi:

```
repeat {
        istruzioni
  if (condition) {break}
}
```

*il salto condizionato, posto alla fine del blocco, consente la una nuova esecuzione del blocco iterativo (istruzioni) nel caso la condizione abbia esito negativo; mentre termina se la condizione è vera per mezzo del costrutto break*

# LINGUAGGIO R

## Generazione di un numero casuale

Per generare un numero casuale si può utilizzare la funzione **sample** così definita

**sample(x, size, replace = FALSE, prob = NULL)**

X: vettore o insieme di numeri (intervallo di scelta)

Size: numero di valori casuali da generare

Replace: se impostato a FALSE non accetta valori randomici ripetuti

Prob: attribuisce un peso alla possibilità che un numero sia estratto (estrazione non uniforme)

**ESEMPIO**

```
x<-sample(1:8, 5, replace=FALSE)
print (x)
[1] 4 1 6 5 3

x<- sample(1:5, 6, replace=TRUE)
print (x)
[1] 2 4 2 2 4 3

sample (c('Batman','Superman'), size=10,
replace=T, prob=c(.80,.20))
"Batman"   "Batman"   "Batman"   "Superman" "Batman"
"Batman"   "Batman"   "Batman"   "Superman" "Batman"
```

# LINGUAGGIO R

## Ripetizione forzata: REPEAT… IF

**GIOCO!** Indovinare un intero compreso tra 0 e 5 determinato a priori dalla macchina mediante una funzione in grado di generare un numero aleatorio

```
#generazione di un numero casuale compreso tra 0 e 5 e inserimento nell'oggetto random
random_num <- sample(0:5,1)
#ciclo di lettura dei dati
repeat {
        #lettura dato inserito dall'utente
        operando<-readline(prompt="Inserire un numero da 0 a 5: ")
        #conversione in valore intero
        operando<-as.integer(operando);
        #terminazione del ciclo se il numero immesso dall'utente è uguale a quello generato
        #casualmente
        if (operando==random_num) {print("Hai indovinato"); break;}
}
```

# LINGUAGGIO R

## Iterazione: FOR (generalità)

- Un linguaggio di programmazione ha un costrutto che realizza una **struttura iterativa controllata** che esegue lo stesso blocco di codice per un numero prestabilito di volte
- In generale la sintassi è:

```
for(indice;condizione; incremento)
                {
                ISTRUZIONI
                }
```

*Il blocco delle istruzioni si ripete fino a quando la* condizione *è TRUE*

# LINGUAGGIO R

## Iterazione: FOR in linguaggio R

- Nel Linguaggio R la sintassi è **for (variabile in vettore)**

dove variabile è un oggetto e vettore specifica gli elementi di un array che devono essere scanditi (e che possono essere utilizzabili anche nel blocco delle istruzioni).

Il ciclo si ripete fino ad esaurimento degli elementi

- In altre parole si esegue un determinato blocco di codice per ogni elemento di una collezione (o vettore)

# LINGUAGGIO R

## Iterazione: FOR esempio

Stampa degli elementi di un vettore

```
#inizializzazione di un vettore di tre elementi
vett_num<-c(3, 7, 42)

indice<-1
for (elem in vett_num) {
  cat("Ciclo ",indice,"-esimo con valore di elem ", elem,"\n")
}

#Visualizzazione script
[1] Ciclo  1 -esimo con valore di elem  3
[1] Ciclo  2 -esimo con valore di elem  7
[1] Ciclo  3 -esimo con valore di elem  42
```

# LINGUAGGIO R

## Iterazione: FOR

- Per manipolare un indice di interi con passo incrementale esplicitato (forma canonica) si può ricorrere alla funzione

**seq(from=**init**, to=**end**, by=**step**)**

che crea un vettore che parte dal valore init fino al valore end con passo incrementale definito da step

**ESEMPIO**

```
v<-seq(from=1,to=3, by=1)
for ( i in v) {
        print(i)
}

#Visualizzazione script
[1] 1
[1] 2
[1] 3
```

# LINGUAGGIO R

## Iterazione: FOR

- La funzione può essere utilizzata per specificare il **passo incrementare** manipolando l'argomento by

**ESEMPIO**

```
v<-seq(from=1,to=5, by=2)
for ( i in v) {
        print(i)
      }
#Visualizzazione script
[1] 1
[1] 3
[1] 5
```

# LINGUAGGIO R

## Iterazione: FOR esempio

### Media di sei elementi

*L'algoritmo impiega un vettore di sei elementi per consentire l'immissione degli operandi da tastiera; finito il ciclo iterativo si svolge la media*

```
#Inizializzazione totalizzatore
tot<-0;
#ciclo di sei passi
for (i in c(1,2,3,4,5,6)) {
#lettura da input del valore e conversione in intero
  operando<-readline(prompt="Inserire un numero: ")
  operando<-as.integer(operando);
#incremento totalizzatore
  tot<-tot+operando;
}
#Calcolo media
media<-tot/6
#Stampa media
print(media)
```

# LINGUAGGIO R

## Iterazione: FOR

- Nel Linguaggio R ha una sua valenza l'istruzione **next** che **salta il passo corrente senza terminare il ciclo**. Quando si incontra next, non si svolge l'ulteriore valutazione e si avvia l'iterazione successiva del ciclo

**ESEMPIO**

```
#Definizione dei succhi
vettore_succhi<- c("mela", "albicocca", "pera",
"kiwi")
for(i in vettore_succhi) {
        #esclusione del succo al gusto di pera
                if (i == "pera") {next}
                at("\nMi piace il succo di:", i)
                }
```

```
Mi piace il succo di: mela
Mi piace il succo di: albicocca
Mi piace il succo di: kiwi
```

# LINGUAGGIO R

## Iterazione: FOR (annidamento)

- Il Linguaggio R consente anche l'**annidamento dell'iterazione**

```
for ( elem in vett){
        istruzioni
        for ( elem in vett){
                istruzioni
        }
        istruzioni
}
```

# LINGUAGGIO R

## Iterazione: FOR (annidamento)

Stampa di un triangolo rettangolo di altezza stabilita dall'utente

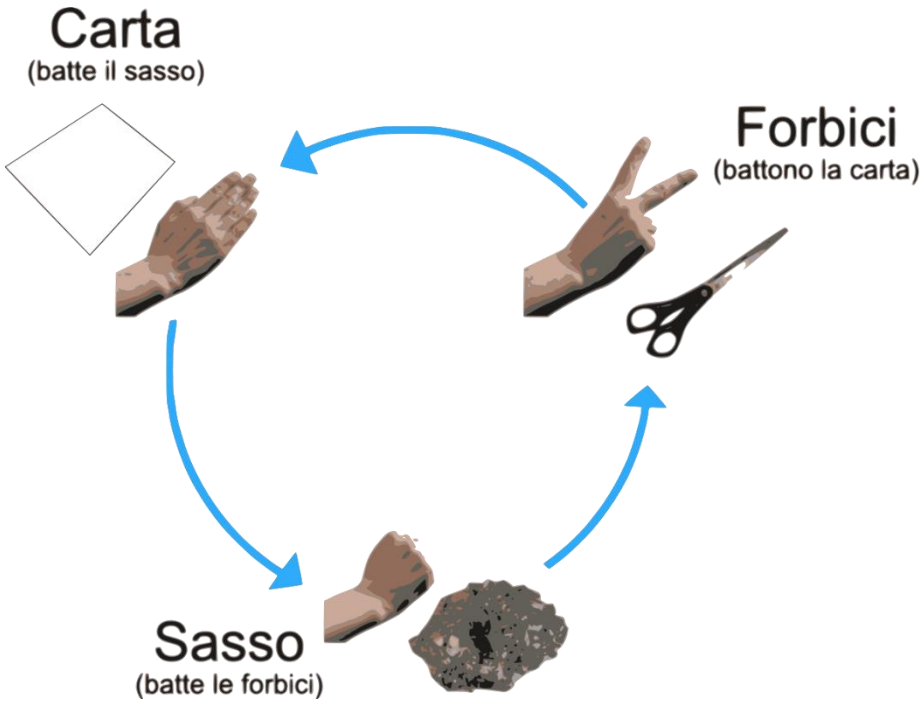
```
print("Inserire la lunghezza della base")
altezza<-scan()
for(i in (1:altezza)){
  for (j in (1:altezza)){
    if ((altezza-(j-1))>i) {
      cat(" ")
    }else { cat("*")}
  }
  cat("\n")
}
```

```
         *
        **
       ***
      ****
     *****
    ******
   *******
  ********
 *********
**********
```

# LINGUAGGIO R

## Iterazione: FOR (annidamento)

GIOCO: LA MORRA CINESE



# LINGUAGGIO R

## Iterazione: FOR (annidamento)

### GIOCO: LA MORRA CINESE (UNA SOLA PARTITA)

```
#Permetto al giocatore di inserire una morra
morra_giocatore<-readline(prompt="\nCarta, Sasso o Forbice? ");
morra_elaboratore=sample(c("Carta", "Sasso","Forbice"),1);
cat("\nScelta elaboratore: ", morra_elaboratore, "Scelta utente: ", morra_giocatore);
punti_giocatore=0;
punti_elaboratore=0;
if (morra_giocatore==morra_elaboratore){prompt("\nMorra Patta")
 }else{
   if ((morra_giocatore=="Carta")&(morra_elaboratore=="Sasso")){punti_giocatore=punti_giocatore+1;
   }else{
   if ((morra_giocatore=="Forbice")&(morra_elaboratore=="Carta")){punti_giocatore=punti_giocatore+1;
   }else {
   if ((morra_giocatore=="Sasso")&(morra_elaboratore=="Forbice")){punti_giocatore=punti_giocatore+1;
   } else{punti_elaboratore=punti_elaboratore+1}
     }
   }
 }
cat("\nPunti Giocatore:",punti_giocatore," Punti Elaboratore:", punti_elaboratore);
```

# LINGUAGGIO R

## Iterazione: FOR (annidamento)

### GIOCO: LA MORRA CINESE (SEI PARTITE)

```
giocate=0; vittoria=FALSE;
punti_giocatore=0; punti_elaboratore=0;
repeat{
morra_giocatore<-readline(prompt="\nCarta, Sasso o Forbice? ");
 morra_elaboratore=sample(c("Carta", "Sasso","Forbice"),1);
 cat("\nScelta elaboratore: ", morra_elaboratore, "Scelta utente: ", morra_giocatore);
 if (morra_giocatore==morra_elaboratore){
 }else{
  if ((morra_giocatore=="Carta")&(morra_elaboratore=="Sasso")){punti_giocatore=punti_giocatore+1;
  }else{
   if ((morra_giocatore=="Forbice")&(morra_elaboratore=="Carta")){punti_giocatore=punti_giocatore+1;
   }else {
    if ((morra_giocatore=="Sasso")&(morra_elaboratore=="Forbice")){punti_giocatore=punti_giocatore+1;
    } else{punti_elaboratore=punti_elaboratore+1
    }
   }
  }
 }
 cat("\nPunti Giocatore:",punti_giocatore," Punti Elaboratore:", punti_elaboratore);
 if ((punti_giocatore>3)|(punti_elaboratore>3)){
  vittoria=TRUE;
  }
 giocate=giocate+1;
 if ((giocate>6)&(vittoria=TRUE)){
  break
 }
}
```

# LINGUAGGIO R

## Iterazione: FOR (annidamento)

### GIOCO: LA MORRA CINESE (SEI PARTITE- TRUCCATO)

```
giocate=0; vittoria=FALSE; punti_giocatore=0; punti_elaboratore=0;
repeat{
morra_giocatore<-readline(prompt="\nCarta, Sasso o Forbice? ");
 morra_elaboratore=sample(c("Carta", "Sasso","Forbice"),1, prob=probabilita);
 cat("\nScelta elaboratore: ", morra_elaboratore, "Scelta utente: ", morra_giocatore);
 if (morra_giocatore==morra_elaboratore){
 }else{
  if ((morra_giocatore=="Carta")&(morra_elaboratore=="Sasso")){punti_giocatore=punti_giocatore+1;
  }else{
   if ((morra_giocatore=="Forbice")&(morra_elaboratore=="Carta")){punti_giocatore=punti_giocatore+1;
   }else {
    if ((morra_giocatore=="Sasso")&(morra_elaboratore=="Forbice")){punti_giocatore=punti_giocatore+1;
    } else{punti_elaboratore=punti_elaboratore+1
    }
   }
  }
 }
 cat("\nPunti Giocatore:",punti_giocatore," Punti Elaboratore:", punti_elaboratore);
 if ((punti_giocatore>3)|(punti_elaboratore>3)){
  vittoria=TRUE;
  }
 giocate=giocate+1;
 if ((giocate>6)&(vittoria=TRUE)){
  break
 }
}
```

```
p_carta=0.33;
 p_sasso=0.33
 p_forbice=0.33;
 if (morra_giocatore=="Carta"){p_carta=0.1;p_sasso=0.1;p_forbice=0.8;}
 if (morra_giocatore=="Sasso"){p_carta=0.8;p_sasso=0.1;p_forbice=0.1;}
 if (morra_giocatore=="Forbice"){p_carta=0.1;p_sasso=0.8;p_forbice=0.1;}
 probabilita=c(p_carta,p_sasso,p_forbice);
```

Fine