IEIM 2018-2019

Esercitazione VIII

"Ripasso Generale"

Alessandro A. Nacci
alessandro.nacci@polimi.it - www.alessandronacci.it

Matrici e funzioni

Esercizio 1

MATRICI E FUNZIONI

Il passaggio di una matrice ad una funzione
e' sempre per indirizzo e mai per copia.

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

```c
void foo1(int *mat)
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  *(mat+i*C+j) = (i+j) * 2;
}
```

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

Puntatore ad intero!

```c
void foo1(int *mat)
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  *(mat+i*C+j) = (i+j) * 2;
}
```

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

```c
void foo1(int *mat)
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  *(mat+i*C+j) = (i+j) * 2;
}
```

Puntatore ad intero!

Calcolo manuale dello spiazzamento

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

```c
void foo1(int *mat)
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  *(mat+i*C+j) = (i+j) * 2;
}
```

Puntatore ad intero!

Calcolo manuale dello spiazzamento

Valore contenuto all'indirizzo tra parentesi

```c
*(mat+i*C+j)
```

INDIRIZZO

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);


int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

```c
void foo2(int mat[R][C])
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  mat[i][j] = (i+j) * 3;
}
```

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);


int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

Esplicito che il puntatore punta al primo elemento di una matrice RxC

```c
void foo2(int mat[R][C])
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  mat[i][j] = (i+j) * 3;
}
```

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

```c
void foo2(int mat[R][C])
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  mat[i][j] = (i+j) * 3;
}
```

Esplicito che il puntatore punta al primo elemento di una matrice RxC

Accesso comodo alla matrice!

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
     int mat[R][C];
     foo1(mat);
     foo2(mat);
     foo3(mat);
}
```

```c
void foo3(int mat[][C])
{
     int i, j;
     for(i=0;i<R;i++)
          for(j=0;j<C;j++)
               mat[i][j] = (i+j) * 4;
}
```

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
     int mat[R][C];
     foo1(mat);
     foo2(mat);
     foo3(mat);
}
```

Esplicito che il puntatore punta al primo elemento di una matrice con C colonne

```c
void foo3(int mat[][C])
{
     int i, j;
     for(i=0;i<R;i++)
          for(j=0;j<C;j++)
               mat[i][j] = (i+j) * 4;
}
```

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

```c
void foo3(int mat[][C])
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  mat[i][j] = (i+j) * 4;
}
```

Esplicito che il puntatore punta al primo elemento di una matrice con C colonne

Accesso comodo alla matrice!

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

```c
void foo3(int mat[][C])
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  mat[i][j] = (i+j) * 4;
}
```

Esplicito che il puntatore punta al primo elemento di una matrice con C colonne

Accesso comodo alla matrice!

NOTA BENE

Ai fini del calcolo dello spiazzamento il numero di righe non è essenziale!

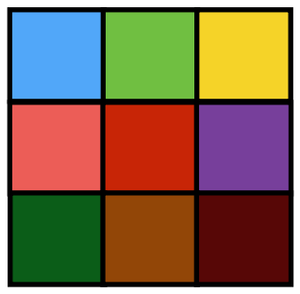
Esercizio 1: Matrici e Funzioni

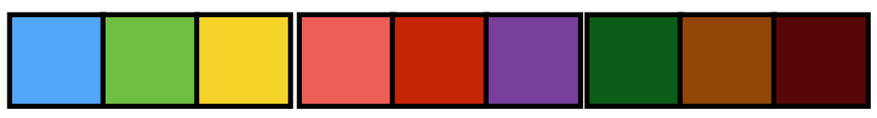
```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

```c
void foo3(int mat[][C])
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  mat[i][j] = (i+j) * 4;
}
```

Esplicito che il puntatore punta al primo elemento di una matrice con C colonne

Accesso comodo alla matrice!





NOTA BENE

Ai fini del calcolo dello spiazzamento il numero di righe non è essenziale!

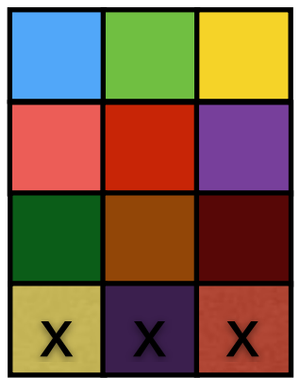
Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

```c
void foo3(int mat[][C])
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  mat[i][j] = (i+j) * 4;
}
```

Esplicito che il puntatore punta al primo elemento di una matrice con C colonne

Accesso comodo alla matrice!



NOTA BENE

Ai fini del calcolo dello spiazzamento il numero di righe non è essenziale!

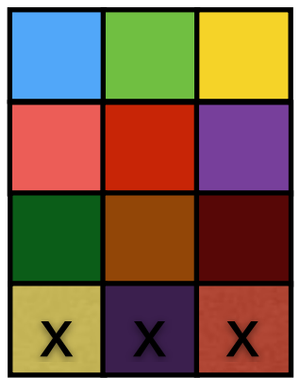
Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

```c
void foo3(int mat[][C])
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  mat[i][j] = (i+j) * 4;
}
```

Esplicito che il puntatore punta al primo elemento di una matrice con C colonne

Accesso comodo alla matrice!



NOTA BENE

Ai fini del calcolo dello spiazzamento il numero di righe non è essenziale!

Esercizio 1: Matrici e Funzioni

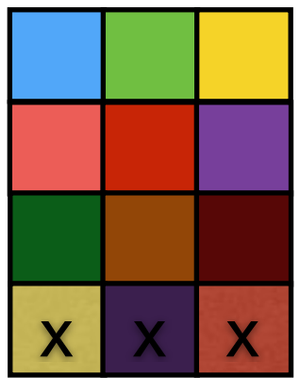
```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

Il valore di mat cambia

Esplicito che il puntatore punta al primo elemento di una matrice con C colonne

```c
void foo3(int mat[][C])
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  mat[i][j] = (i+j) * 4;
}
```

Accesso comodo alla matrice!



NOTA BENE

Ai fini del calcolo dello spiazzamento il numero di righe non è essenziale!

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

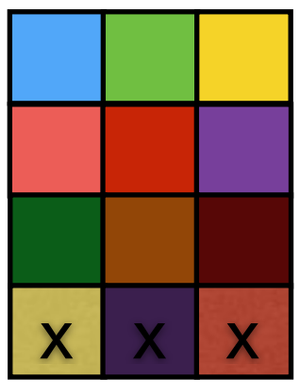
Il valore di mat cambia

Il valore di mat cambia

```c
void foo3(int mat[][C])
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  mat[i][j] = (i+j) * 4;
}
```

Esplicito che il puntatore punta al primo elemento di una matrice con C colonne

Accesso comodo alla matrice!



NOTA BENE

Ai fini del calcolo dello spiazzamento il numero di righe non è essenziale!

Esercizio 1: Matrici e Funzioni

```c
#include <stdio.h>

#define R 3
#define C 3

void foo1(int*);
void foo2(int mat[R][C]);
void foo3(int mat[][C]);

int main()
{
      int mat[R][C];
      foo1(mat);
      foo2(mat);
      foo3(mat);
}
```

Il valore di mat cambia

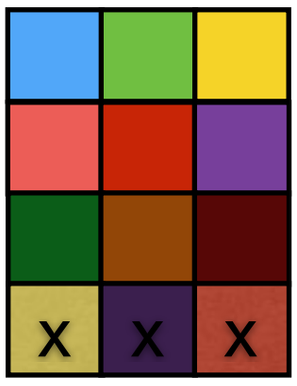
Il valore di mat cambia

Il valore di mat cambia

Esplicito che il puntatore punta al primo elemento di una matrice con C colonne

```c
void foo3(int mat[][C])
{
      int i, j;
      for(i=0;i<R;i++)
            for(j=0;j<C;j++)
                  mat[i][j] = (i+j) * 4;
}
```

Accesso comodo alla matrice!



NOTA BENE

Ai fini del calcolo dello spiazzamento il numero di righe non è essenziale!

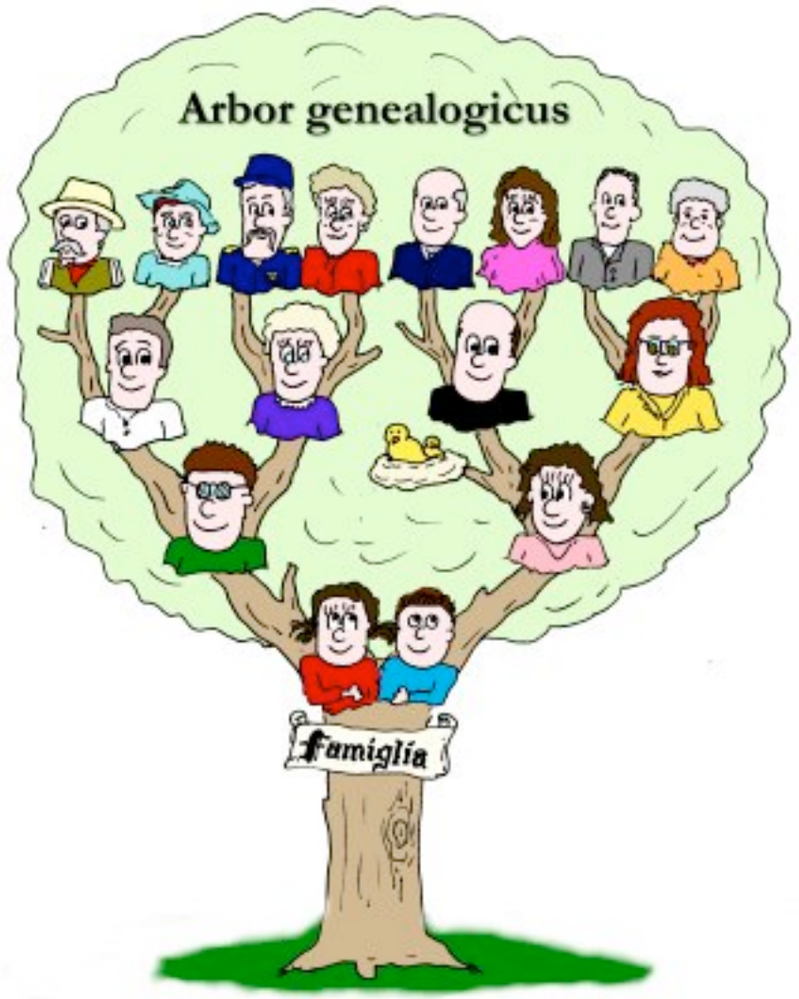
L’albero genealogico

Esercizio 2

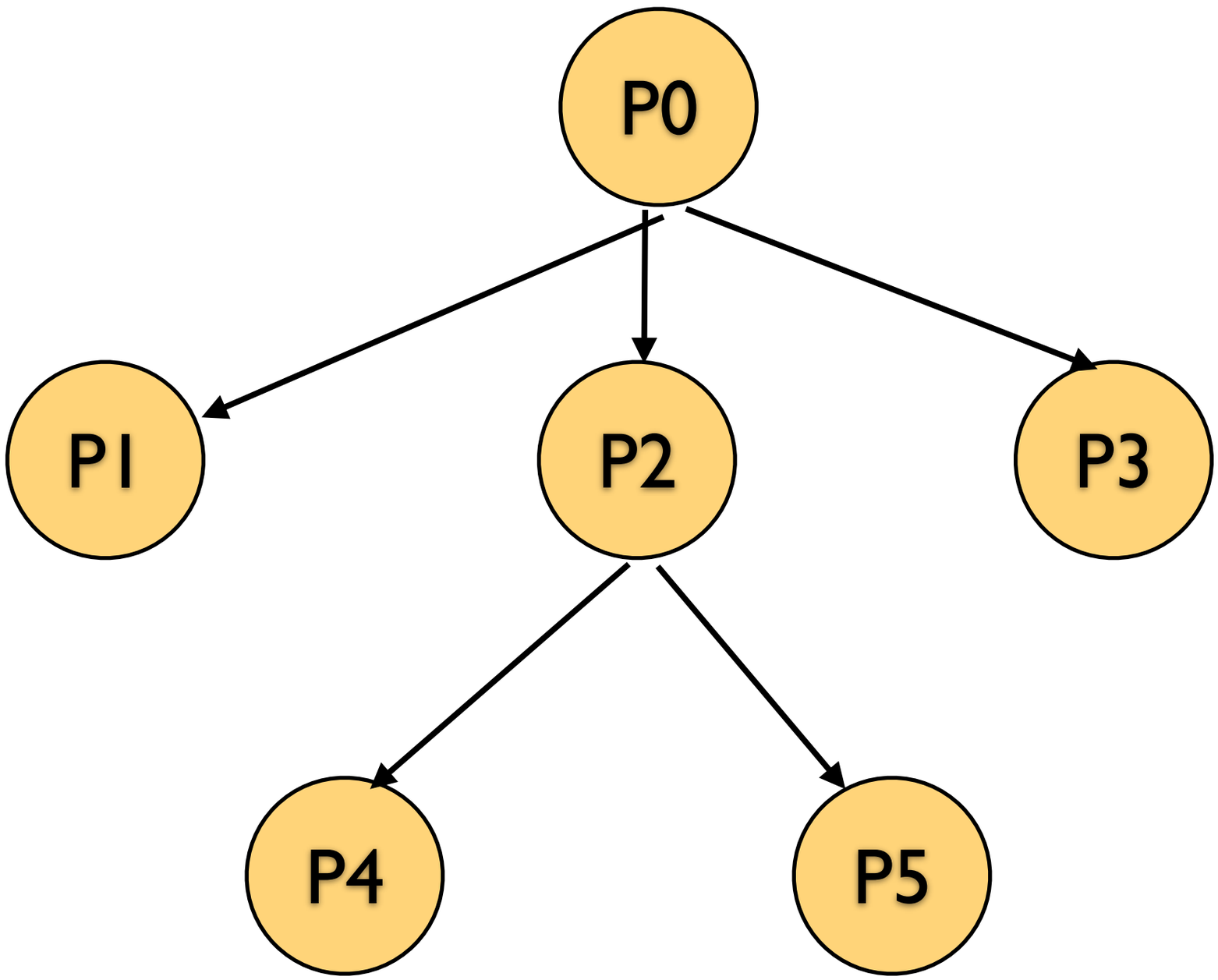
L’albero genealogico

- Scrivere un programma C che sia in grado di rappresentare e gestire un albero genealogico

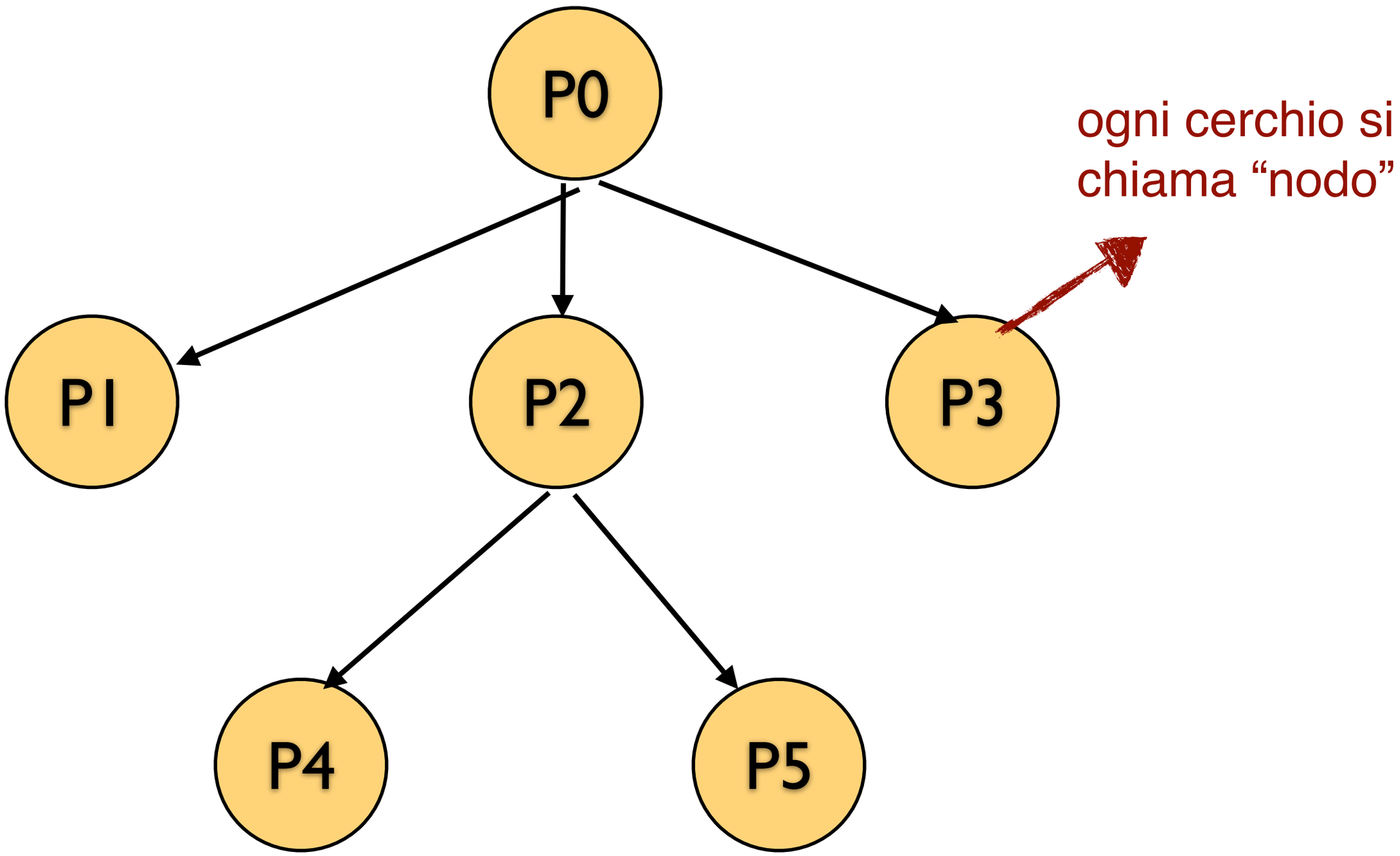
- In particolare, vogliamo poter fare:
  - Creare una persona
  - Rappresentare di una popolazione
  - Aggiungere figli ad una persona
  - Elencare i figli e i nipoti dato un antenato



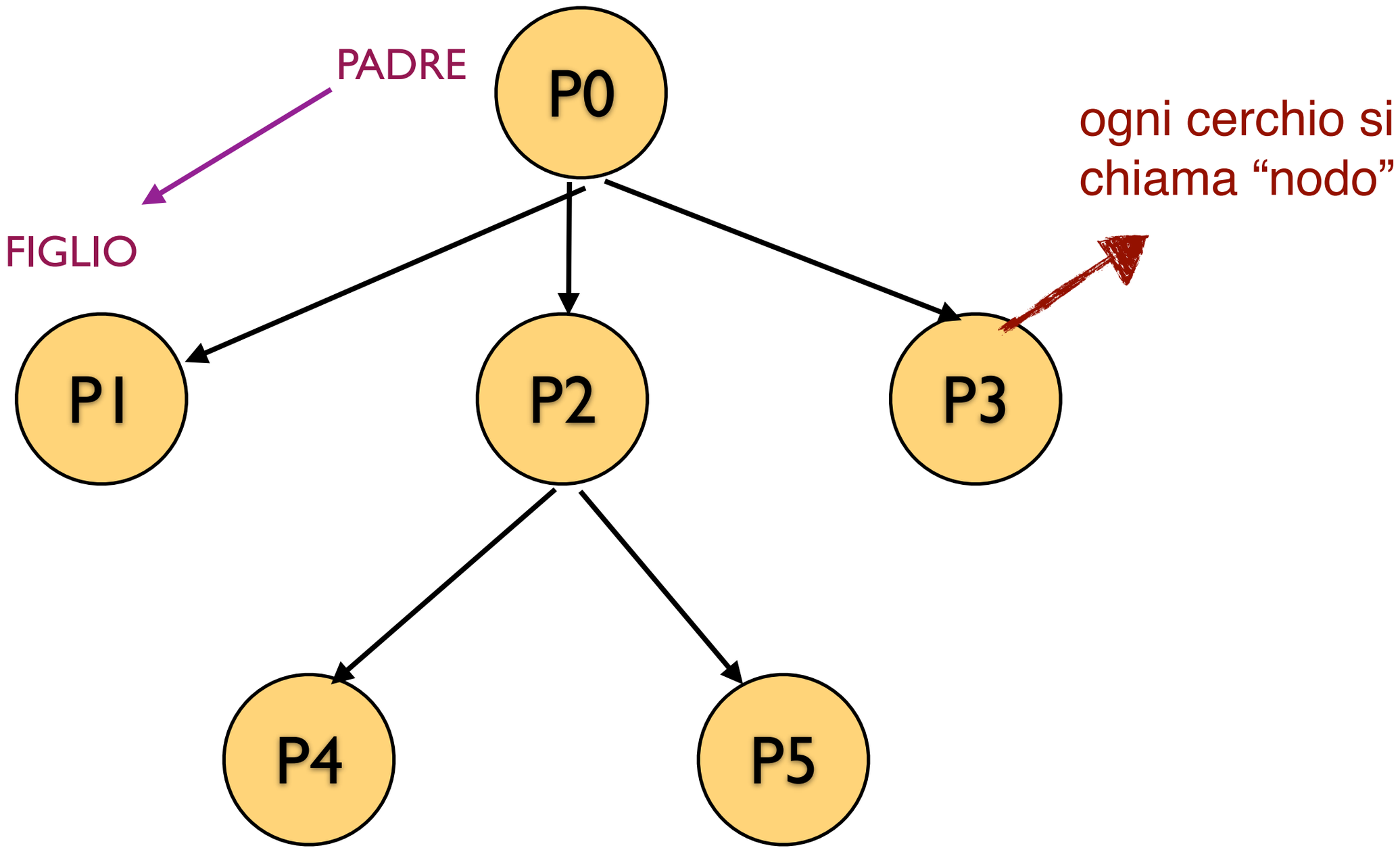
Una famosa struttura dati: l'albero



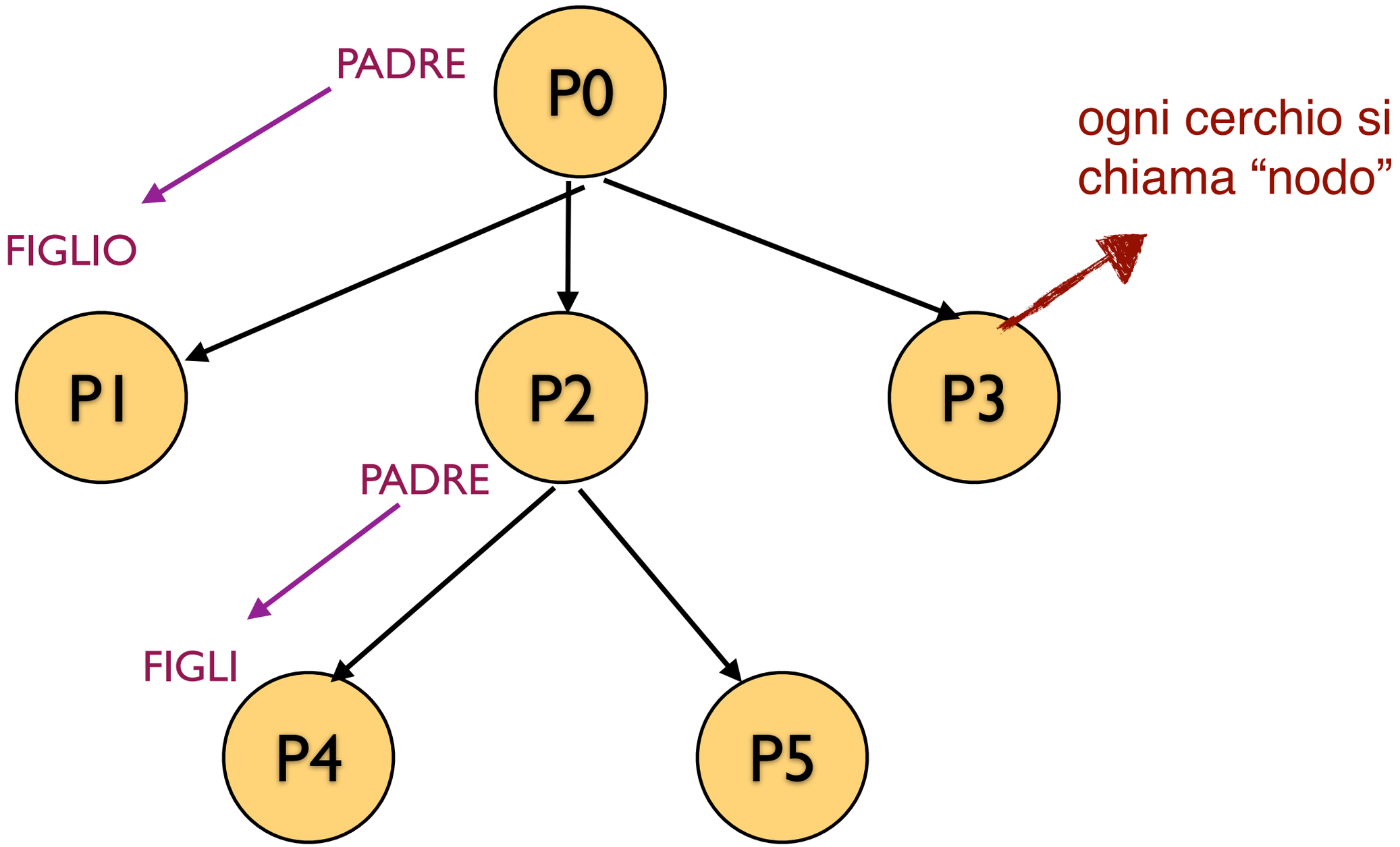
Una famosa struttura dati: l'albero



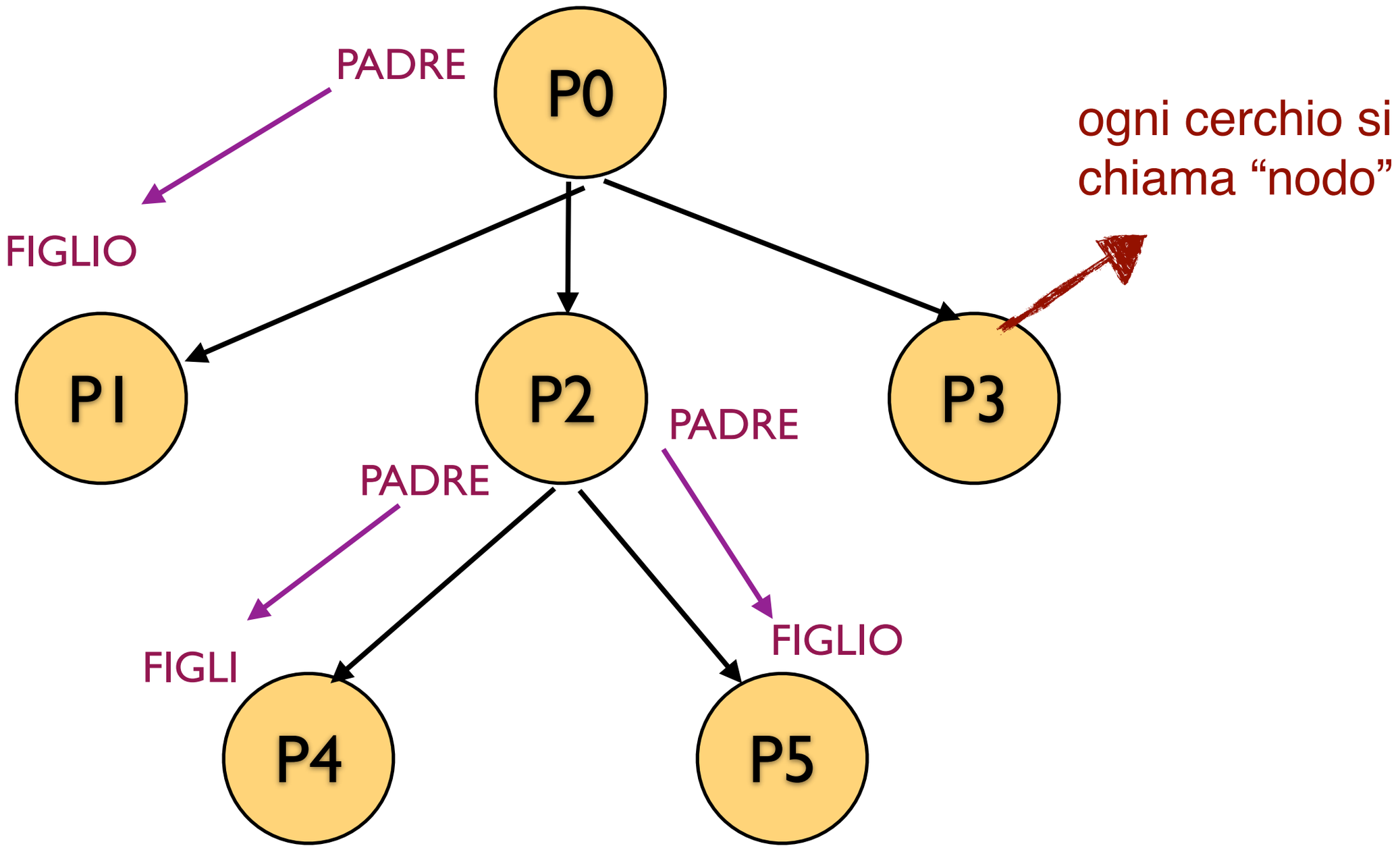
Una famosa struttura dati: l'albero



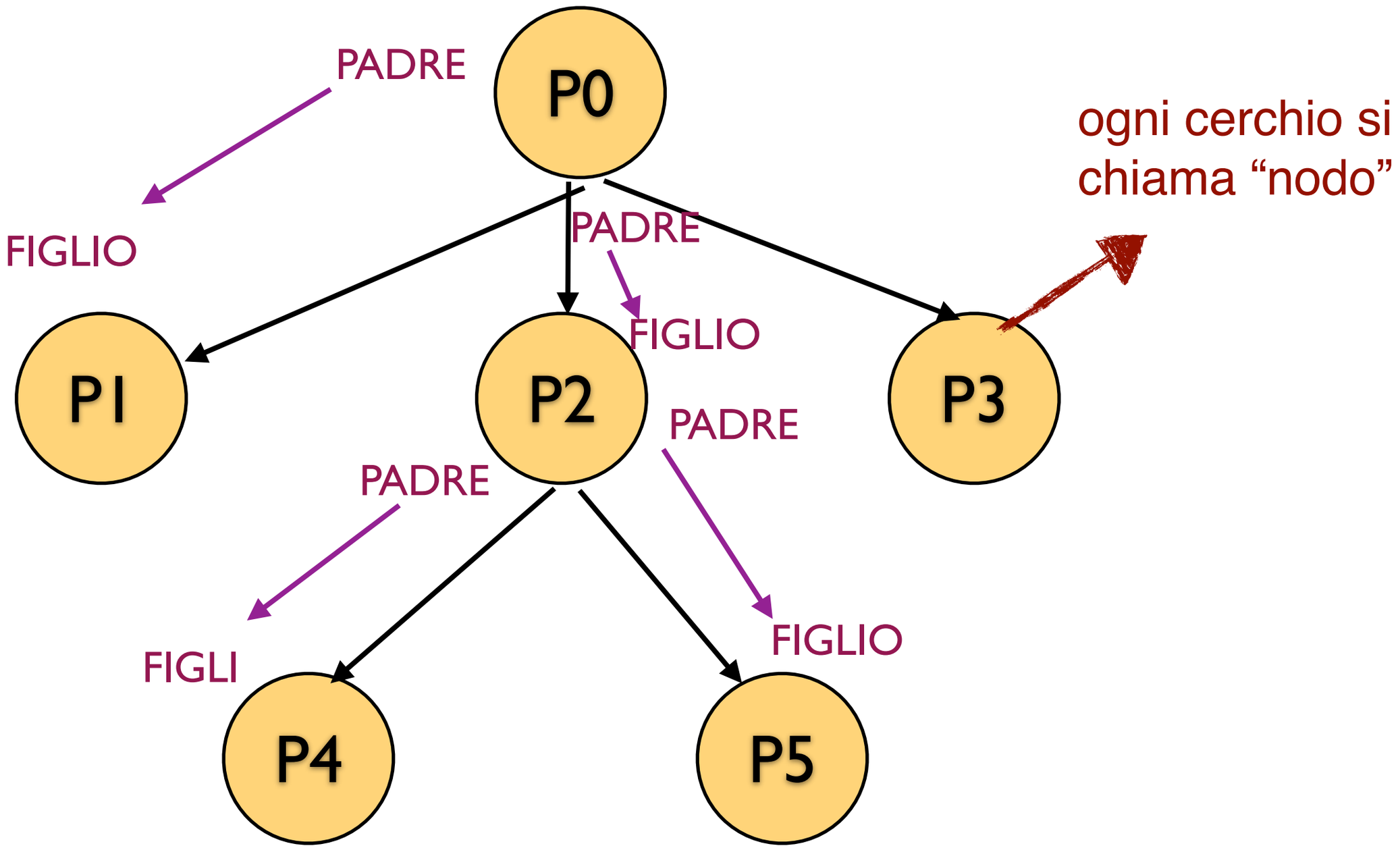
Una famosa struttura dati: l'albero



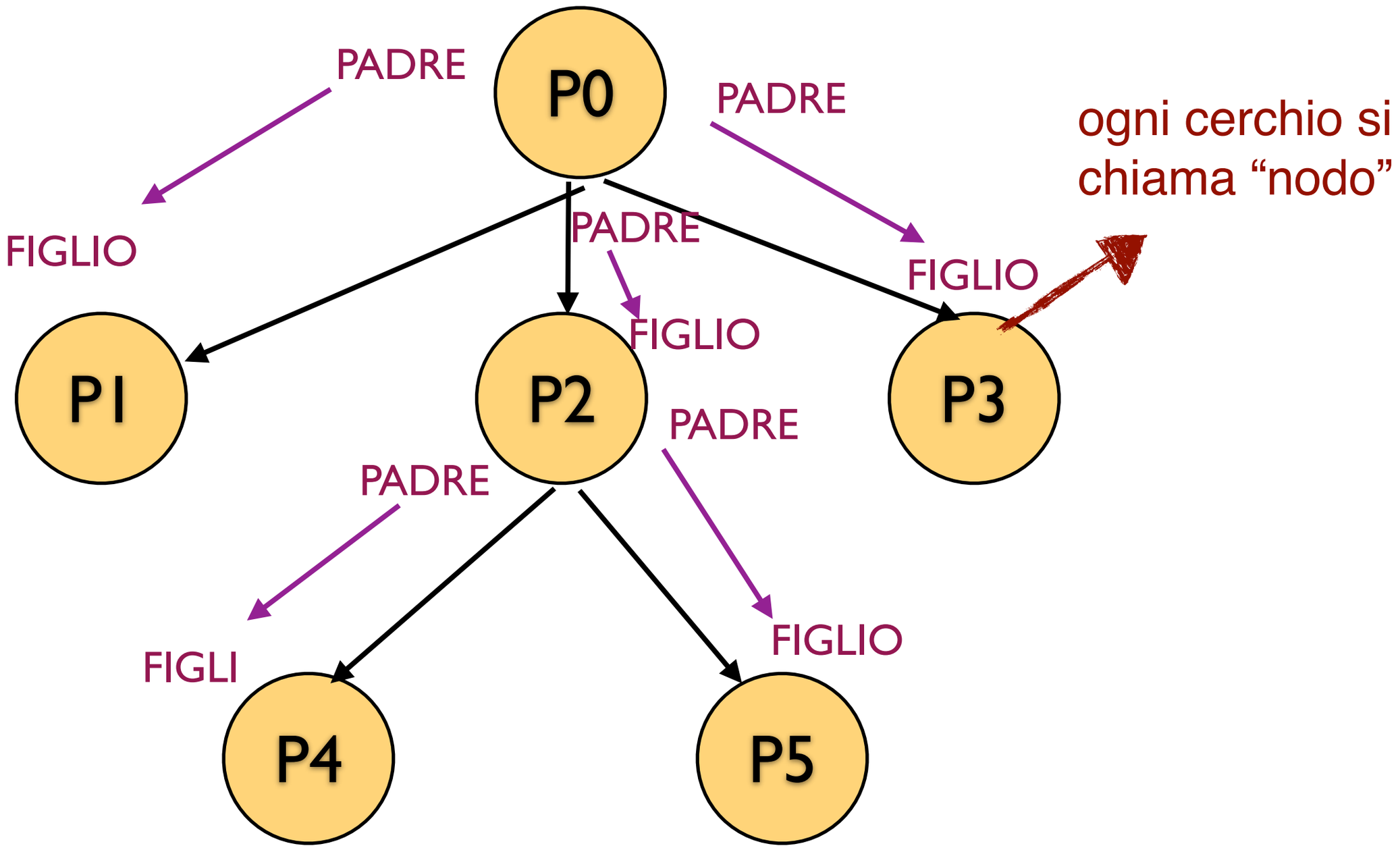
Una famosa struttura dati: l'albero



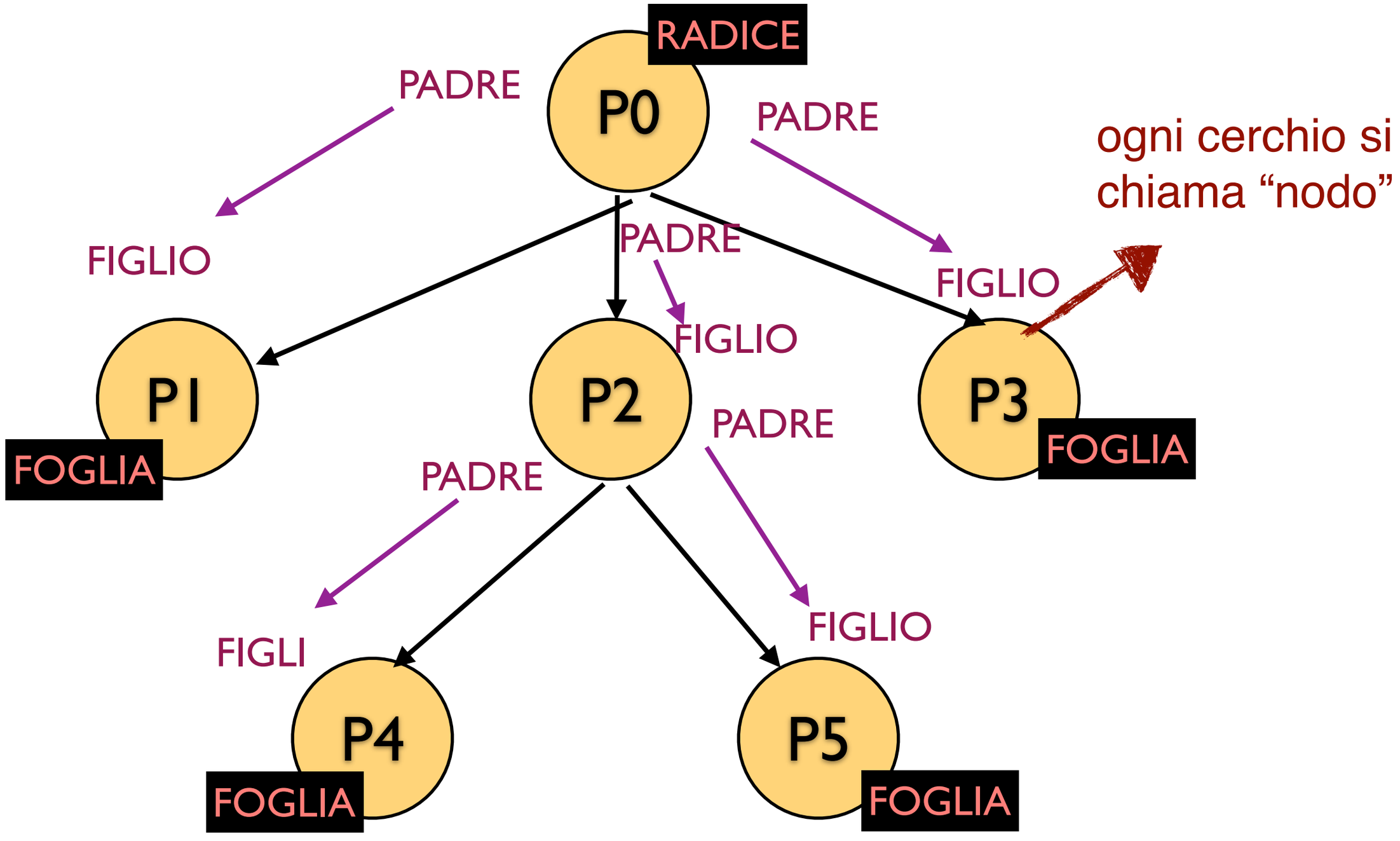
Una famosa struttura dati: l'albero



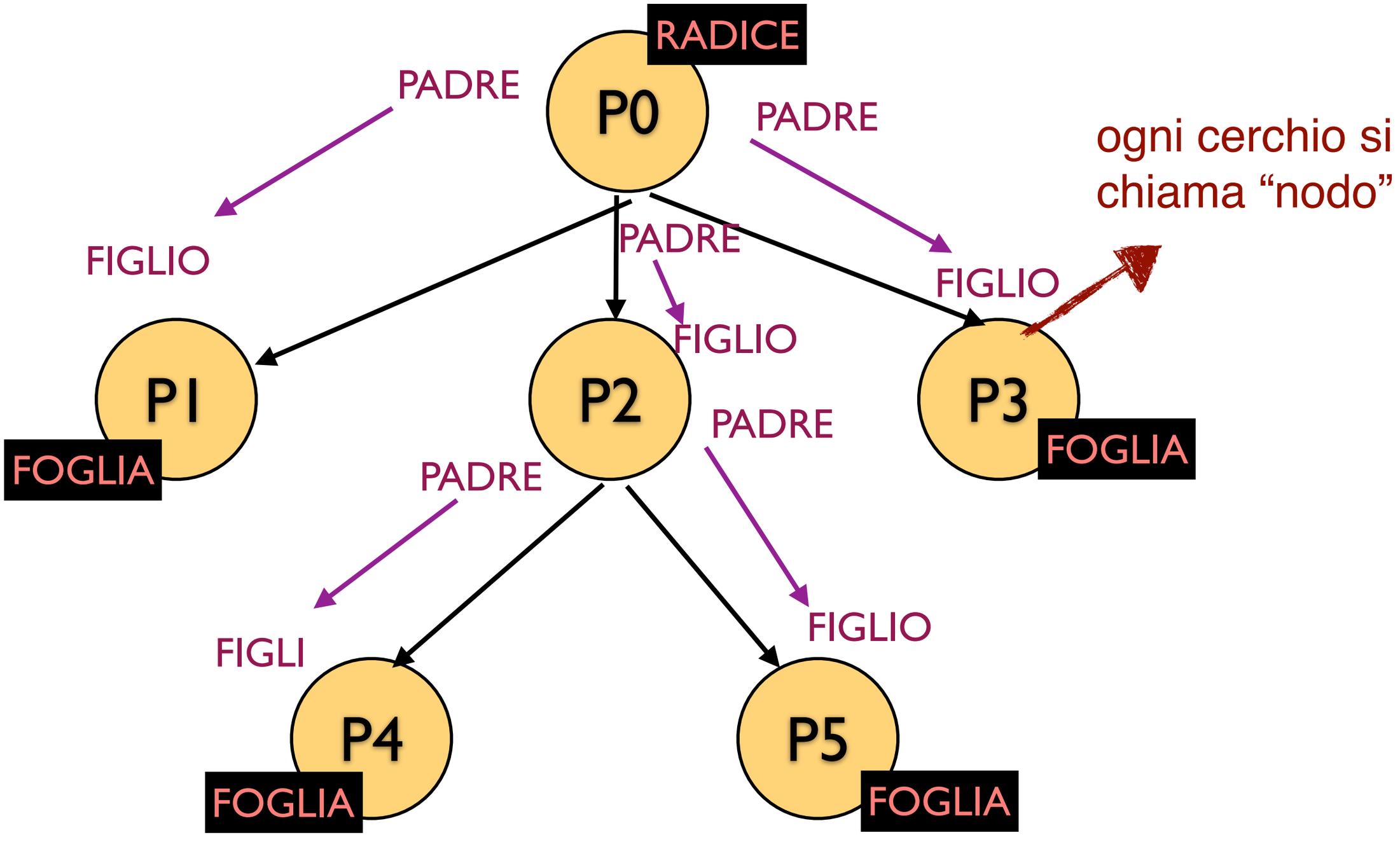
Una famosa struttura dati: l'albero



Una famosa struttura dati: l'albero

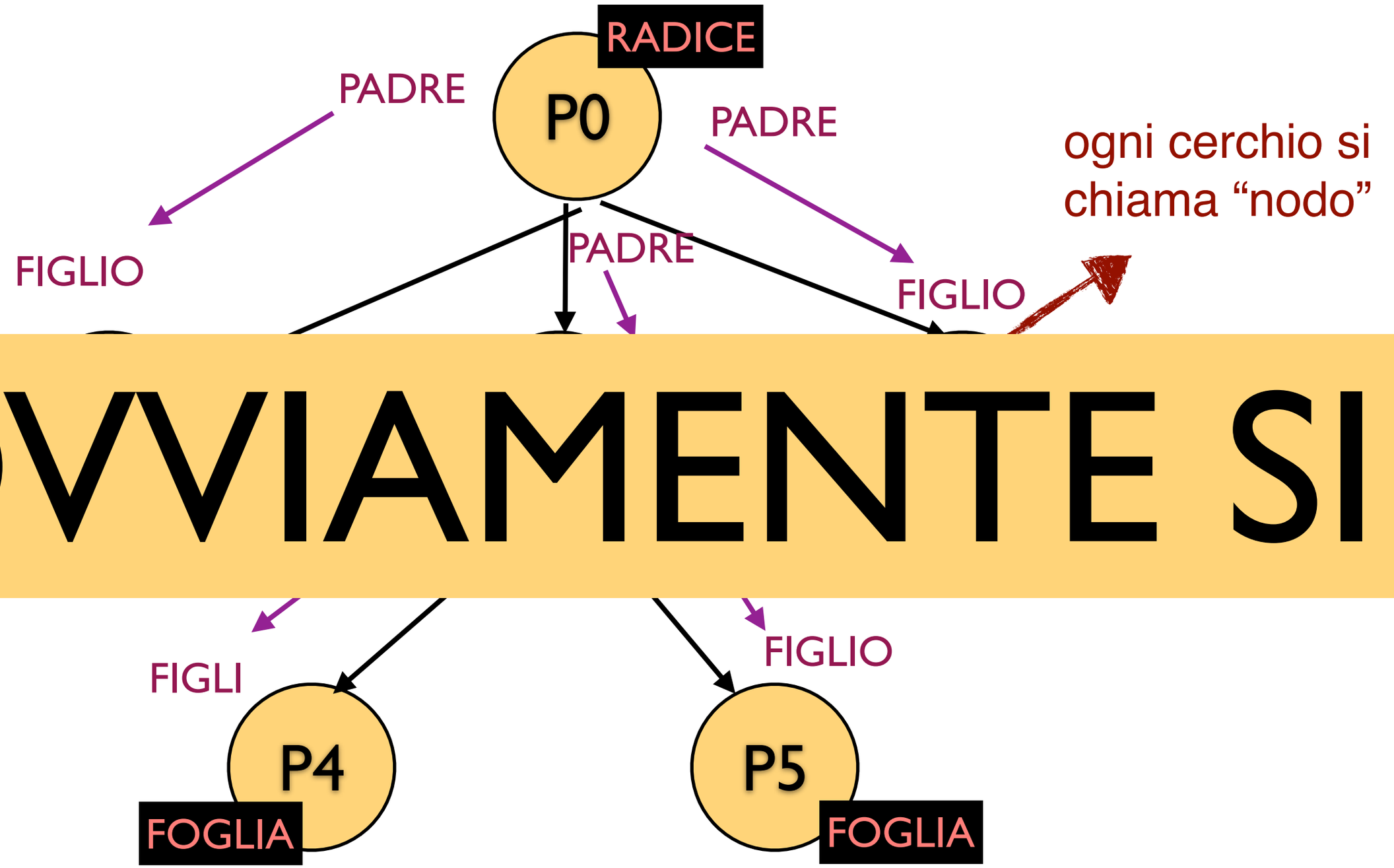


Una famosa struttura dati: l'albero



Può essere utile per rappresentare un albero genealogico?

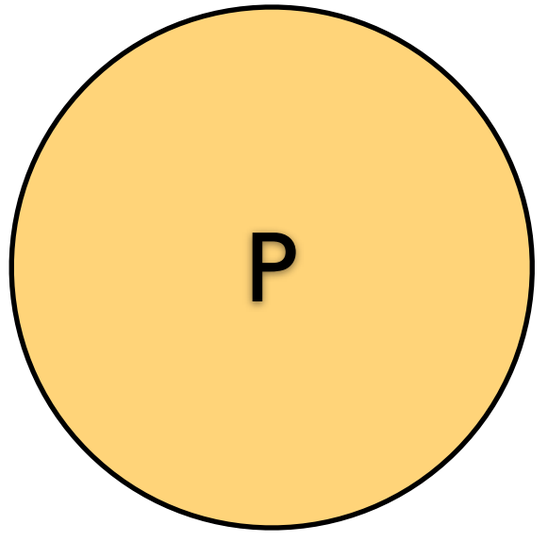
Una famosa struttura dati: l'albero



Può essere utile per rappresentare un albero genealogico?

- OGNI NODO DELL'ALBERO SARA' PER NOI UNA PERSONA

==

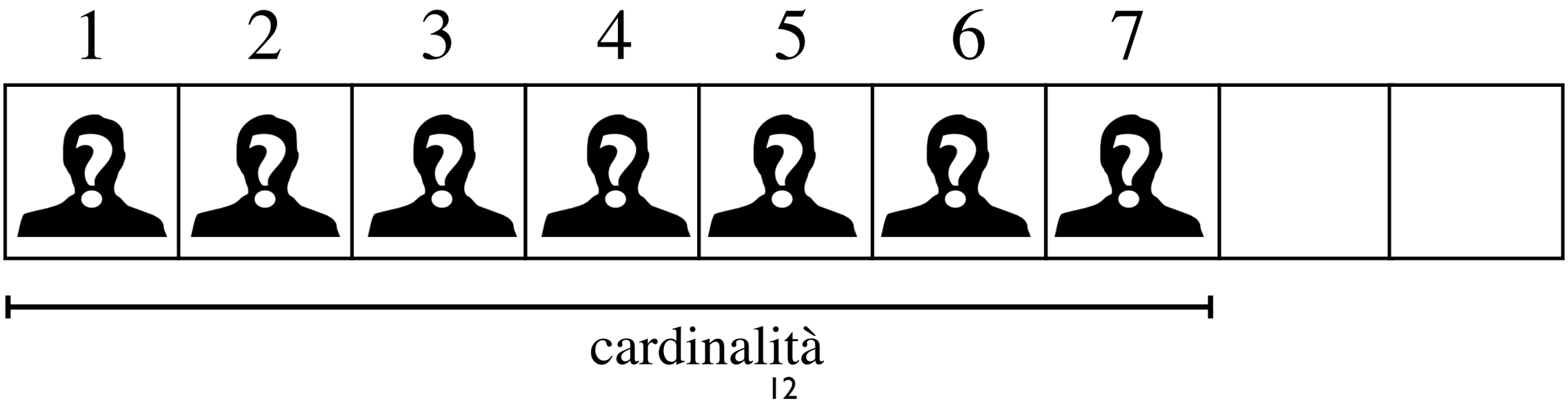


Una Persona

- SESSO
- NOME
- ETA?
- CHI SONO I GENITORI?
- CHI SONO I FIGLI?
- QUANTI FIGLI?

Una Popolazione

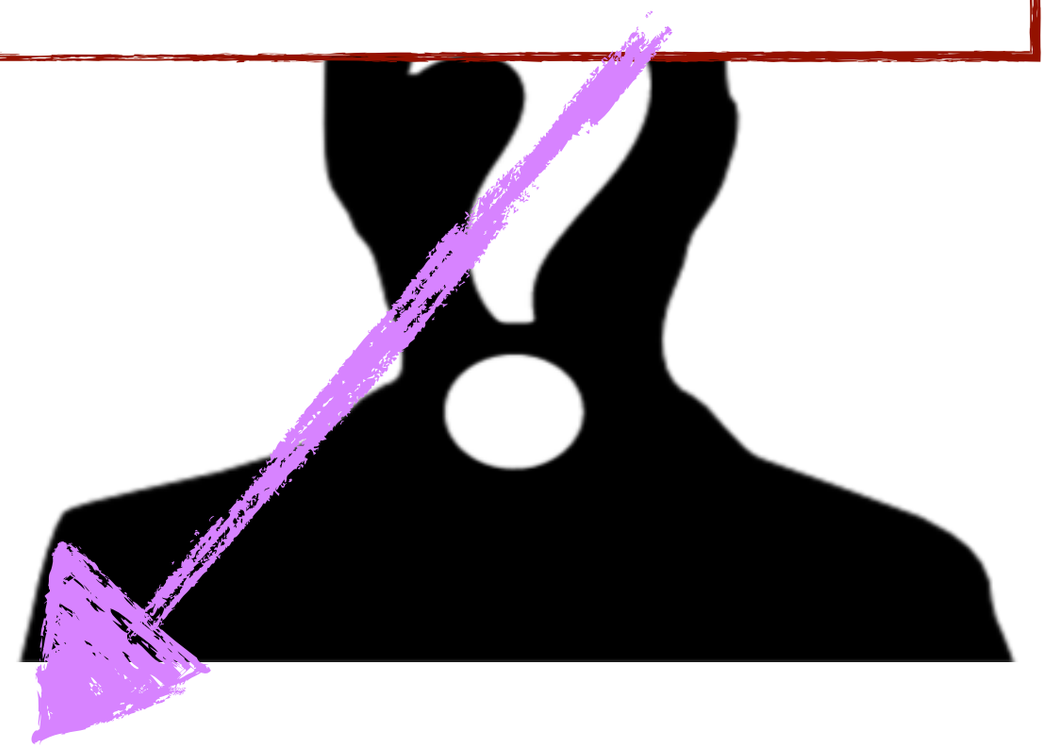
- Una popolazione è rappresentata da un insieme di persone
- Ogni persona ha un suo indice (numero univoco di identificazione)
- Esiste un numero di persone della

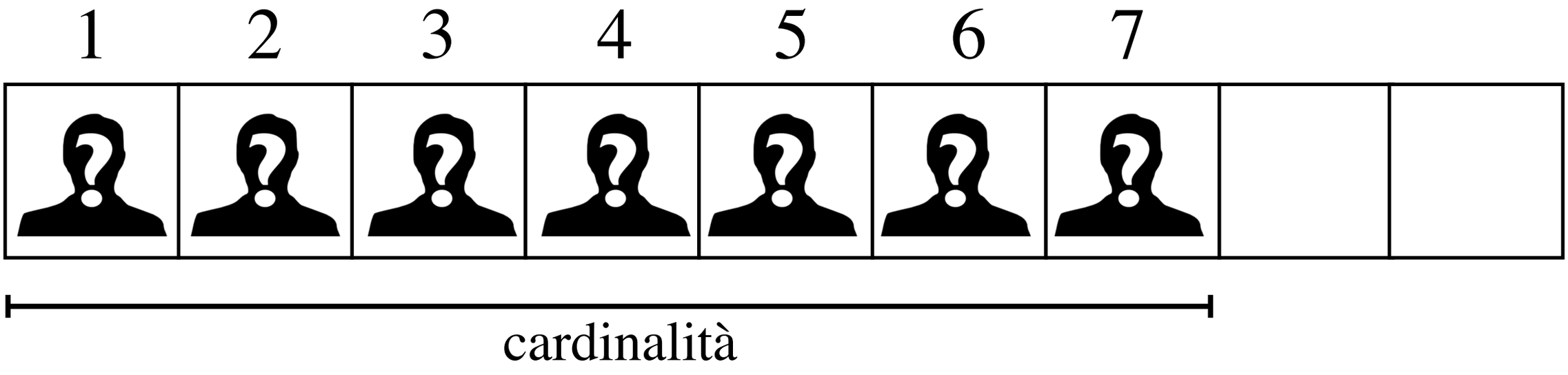


Una Persona nella popolazione

- SESSO
- NOME
- ETA?
- CHI SONO I GENITORI?
- CHI SONO I FIGLI?
- QUANTI FIGLI?

Li rappresentiamo con l'indice della persona nella popolazione





```c
typedef struct {

} persona;
```

```c
int main () {

    persona* popolazione[MAX_PERSONE];
    int cardinalita_popolazione = -1;
```

```
typedef struct {
    genere sesso;
    char nome[STR_LEN];
    int i_persona;
    int i_genitore1;
    int i_genitore2;
    int i_figli[MAX_FIGLI];
    int numero_figli;
    int eta;
} persona;
```

```
int main () |
{

    persona* popolazione[MAX_PERSONE];
    int cardinalita_popolazione = -1;
```

```c
typedef enum {MASCHIO, FEMMINA} genere;

typedef struct {
    genere sesso;
    char nome[STR_LEN];
    int i_persona;
    int i_genitore1;
    int i_genitore2;
    int i_figli[MAX_FIGLI];
    int numero_figli;
    int eta;
} persona;
```

```c
int main () |
{

    persona* popolazione[MAX_PERSONE];
    int cardinalita_popolazione = -1;
```

Creazione di una persona

```
persona crea_persona(genere sesso, char nome[STR_LEN], int eta)
{

}
```

Creazione di una persona

```
persona crea_persona(genere sesso, char nome[STR_LEN], int eta)
{
    persona p;
    p.sesso = sesso;
    strcpy(p.nome,nome);
    p.i_genitore1 = -1;
    p.i_genitore2 = -1;
    p.numero_figli = 0;
    p.eta = eta;
    return p;
}
```

Aggiunta persona alla popolazione

```
int aggiungi_a_popolazione(persona* popolazione[MAX_PERSONE],
    persona* p, int* cardinalita_popolazione)
{

}
```

Aggiunta persona alla popolazione

```
int aggiungi_a_popolazione(persona* popolazione[MAX_PERSONE],
    persona* p, int* cardinalita_popolazione)
{
    (*cardinalita_popolazione)++;
    p->i_persona = *cardinalita_popolazione;
    popolazione[*cardinalita_popolazione] = p;
}
```

Aggiunta di un figlio

```
void aggiungi_figlio(persona* genitore, persona* figlio)
{
}
```

Aggiunta di un figlio

```
void aggiungi_figlio(persona* genitore, persona* figlio)
{
    genitore->i_figli[genitore->numero_figli] = figlio->i_persona;
    genitore->numero_figli++;

    if (figlio->i_genitore1 == -1)
        figlio->i_genitore1 = genitore->i_persona;
    else if (figlio->i_genitore2 == -1)
        figlio->i_genitore2 = genitore->i_persona;
    else
        printf("errore\n");

}
```

Funzioni di stampa a schermo

```
char* genere_to_str(genere sesso)
{



}

void stampa_persona(persona* p)
{


}
```

Funzioni di stampa a schermo

```
char* genere_to_str(genere sesso)
{
    if (sesso == MASCHIO) return "MASCHIO";
    if (sesso == FEMMINA) return "FEMMINA";
    return "?";
}

void stampa_persona(persona* p)
{


}
```

Funzioni di stampa a schermo

```
char* genere_to_str(genere sesso)
{
    if (sesso == MASCHIO) return "MASCHIO";
    if (sesso == FEMMINA) return "FEMMINA";
    return "?";
}

void stampa_persona(persona* p)
{
    printf("Nome: %s - Genere: %s - Eta':%d - Id:%d - #Figli: %d\n",
    p->nome, genere_to_str(p->sesso), p->eta, p->i_persona, p->numero_figli);
}
```

Elenco dei figli e dei nipoti

```
void elenca_figli_nipoti(persona* popolazione[MAX_PERSONE],
    persona* p, genere sesso, int eta_minima)
{

}
```

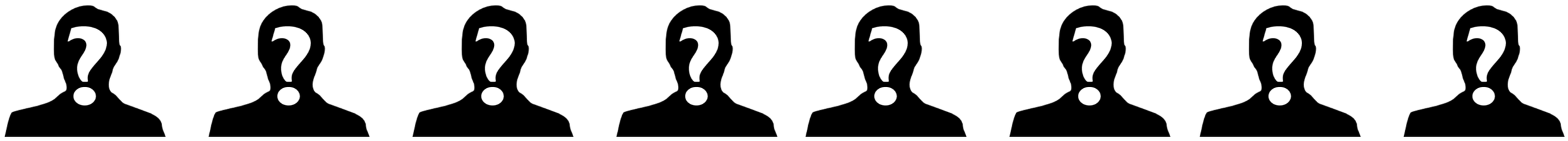
Elenco dei figli e dei nipoti

```
void elenca_figli_nipoti(persona* popolazione[MAX_PERSONE],
    persona* p, genere sesso, int eta_minima)
{

    int i;

    if (p->sesso == sesso && p->eta >= eta_minima)
        stampa_persona(p);


    for (i = 0; i < p->numero_figli; i++)
    {
        persona* f = popolazione[p->i_figli[i]];
        elenca_figli_nipoti(popolazione, f, sesso, eta_minima);
    }
}
```

La nostra popolazione

MARCO	STEFANIA	LUCA	PIPPO	LUCIA	ARIANNA	RINALDO	STEFANO
P0	P1	P2	P3	P4	P5	P6	P7

La nostra popolazione



MARCO P0 | STEFANIA P1 | LUCA P2 | PIPPO P3 | LUCIA P4 | ARIANNA P5 | RINALDO P6 | STEFANO P7

Marco e' padre di LUCA e di PIPPO

Stefania e' madre di LUCA e di PIPPO

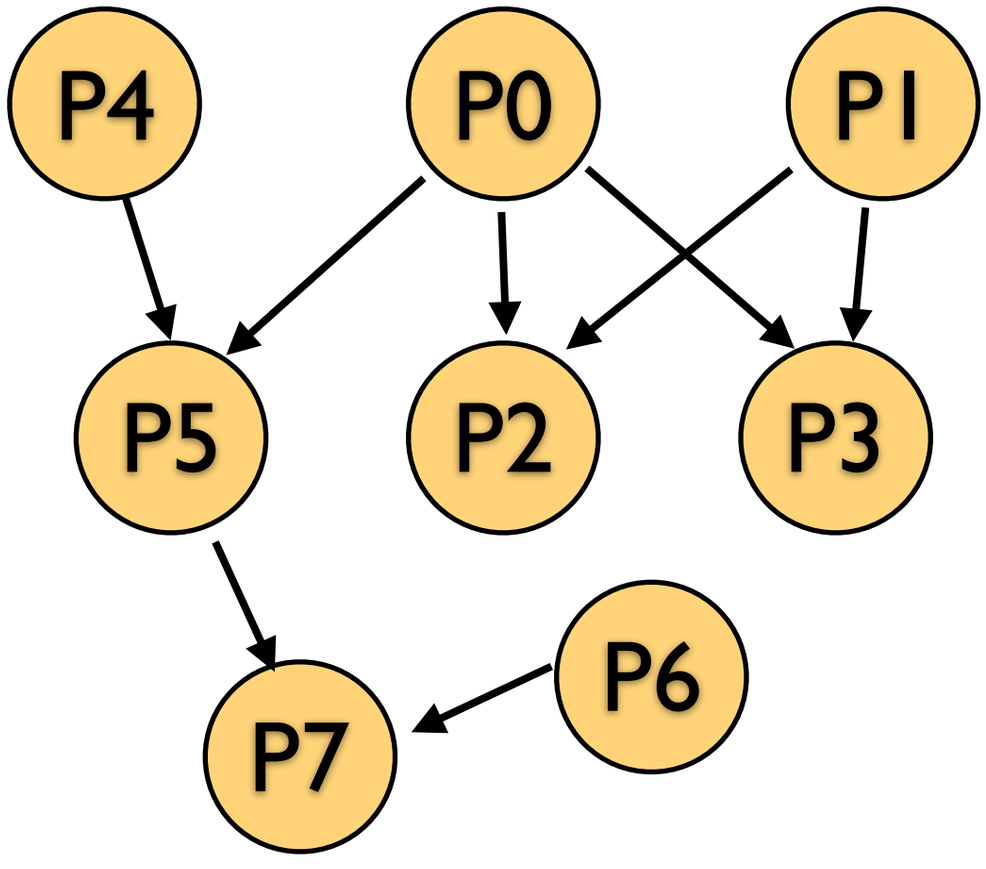
Arianna e' figlia di Marco e Lucia

Stefano e' figlio di Arianna e Rinaldo

La nostra popolazione

MARCO P0 | STEFANIA P1 | LUCA P2 | PIPPO P3 | LUCIA P4 | ARIANNA P5 | RINALDO P6 | STEFANO P7

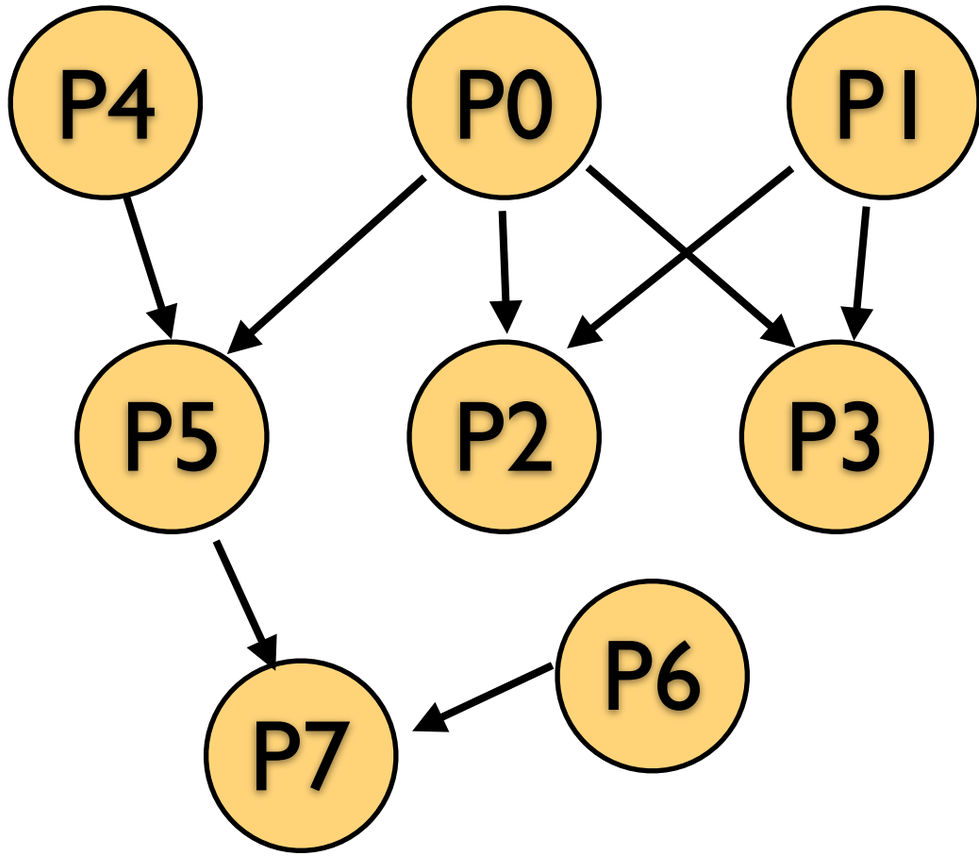
Marco e' padre di LUCA e di PIPPO
Stefania e' madre di LUCA e di PIPPO
Arianna e' figlia di Marco e Lucia
Stefano e' figlio di Arianna e Rinaldo



La nostra popolazione (codice C)

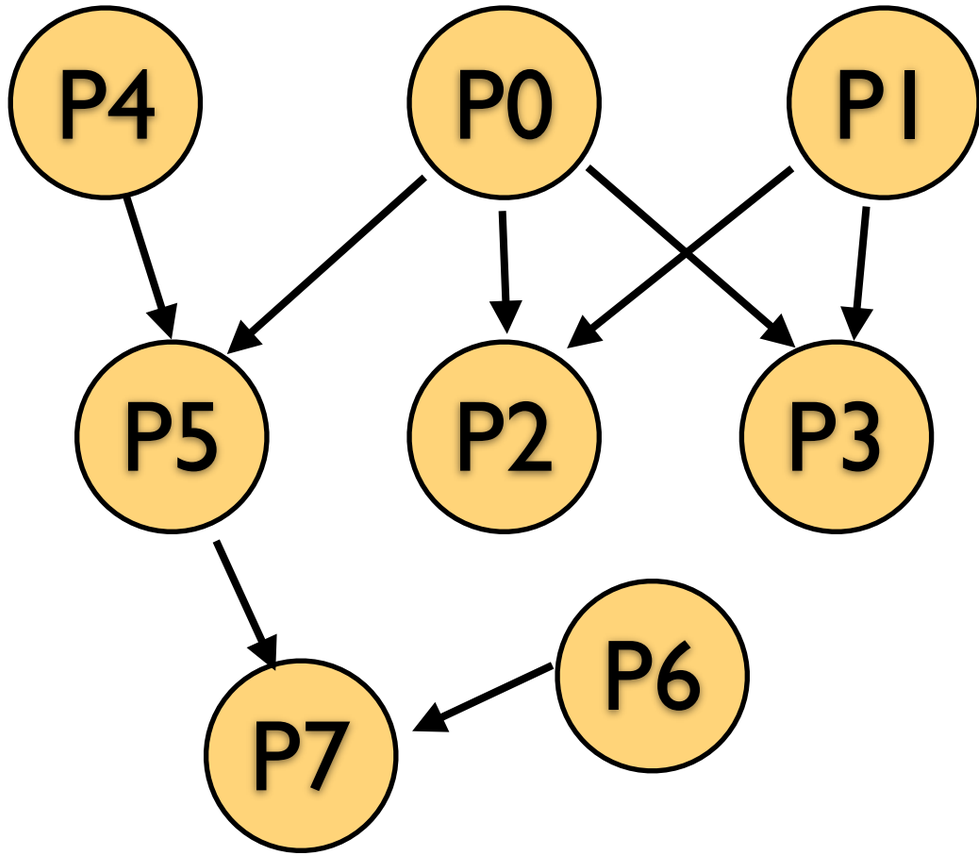
```c
int main ()
{

    persona* popolazione[MAX_PERSONE];
    int cardinalita_popolazione = -1;

    persona p0 = crea_persona(MASCHIO, "MARCO", 50);
    persona p1 = crea_persona(FEMMINA, "STEFANIA", 49);
    persona p2 = crea_persona(MASCHIO, "LUCA", 30);
    persona p3 = crea_persona(MASCHIO, "PIPPO", 26);
    persona p4 = crea_persona(FEMMINA, "LUCIA", 53);
    persona p5 = crea_persona(FEMMINA, "ARIANNA", 30);
    persona p6 = crea_persona(MASCHIO, "RINALDO", 32);
    persona p7 = crea_persona(MASCHIO, "STEFANO", 10);

    aggiungi_a_popolazione(popolazione, &p0, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p1, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p2, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p3, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p4, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p5, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p6, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p7, &cardinalita_popolazione);
```

Aggiungiamo le parentele



Aggiungiamo le parentele

```
// Marco e' padre di LUCA e di PIPPO
aggiungi_figlio(popolazione[0], popolazione[2]);
aggiungi_figlio(popolazione[0], popolazione[3]);

// Stefania e' madre di LUCA e di PIPPO
aggiungi_figlio(popolazione[1], popolazione[2]);
aggiungi_figlio(popolazione[1], popolazione[3]);

// Arianna e' figlia di Marco e Lucia
aggiungi_figlio(popolazione[0], popolazione[5]);
aggiungi_figlio(popolazione[4], popolazione[5]);

// Stefano e' figlio di Arianna e Rinaldo
aggiungi_figlio(popolazione[5], popolazione[7]);
aggiungi_figlio(popolazione[6], popolazione[7]);
```



Il main()

```c
int main ()
{

    persona* popolazione[MAX_PERSONE];
    int cardinalita_popolazione = -1;

    persona p0 = crea_persona(MASCHIO, "MARCO", 50);
    persona p1 = crea_persona(FEMMINA, "STEFANIA", 49);
    persona p2 = crea_persona(MASCHIO, "LUCA", 30);
    persona p3 = crea_persona(MASCHIO, "PIPPO", 26);
    persona p4 = crea_persona(FEMMINA, "LUCIA", 53);
    persona p5 = crea_persona(FEMMINA, "ARIANNA", 30);
    persona p6 = crea_persona(MASCHIO, "RINALDO", 32);
    persona p7 = crea_persona(MASCHIO, "STEFANO", 10);

    aggiungi_a_popolazione(popolazione, &p0, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p1, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p2, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p3, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p4, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p5, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p6, &cardinalita_popolazione);
    aggiungi_a_popolazione(popolazione, &p7, &cardinalita_popolazione);

    // Marco e' padre di LUCA e di PIPPO
    aggiungi_figlio(popolazione[0], popolazione[2]);
    aggiungi_figlio(popolazione[0], popolazione[3]);

    // Stefania e' madre di LUCA e di PIPPO
    aggiungi_figlio(popolazione[1], popolazione[2]);
    aggiungi_figlio(popolazione[1], popolazione[3]);

    // Arianna e' figlia di Marco e Lucia
    aggiungi_figlio(popolazione[0], popolazione[5]);
    aggiungi_figlio(popolazione[4], popolazione[5]);

    // Stefano e' figlio di Arianna e Rinaldo
    aggiungi_figlio(popolazione[5], popolazione[7]);
    aggiungi_figlio(popolazione[6], popolazione[7]);


    elenca_figli_nipoti(popolazione, popolazione[0], MASCHIO, 3);

    return 0;

}
```

Tutte il materiale sarà disponibile sul mio sito internet!

alessandronacci.it