Corso di Informatica Medica

Esercitazione VIII

Alessandro A. Nacci
nacci@elet.polimi.it - alessandronacci.com

READY...
SET...
GO

Lancio dei dati

Data la sequenza di lanci di un dato che ha prodotto il seguente risultato:

{3,4,6,1,3,5,4,2,6,2,3}

- Calcolare media e mediana degli elementi di un array contenente i lanci di un dado.
- Calcolare inoltre le frequenze di ogni valore.

La media

Dato che sappiamo usare le sommatorie... $M_a = \frac{1}{n} \sum_{i=1}^{n} x_i$

```
//Funzione che calcola il valore medio di un array
void calcolaMedia(int array[], int *media){

}
```

La media

Dato che sappiamo usare le sommatorie... $M_a = \frac{1}{n}\sum_{i=1}^{n} x_i$

```
//Funzione che calcola il valore medio di un array
void calcolaMedia(int array[], int *media){
    int i;
    for(i=0;i<N;i++)
    {
        *media = *media + array[i];
    }

    *media = *media/N;
}
```

La media

Dato che sappiamo usare le sommatorie... $M_a = \frac{1}{n}\sum_{i=1}^{n} x_i$

```
//Funzione che calcola il valore medio di un array
void calcolaMedia(int array[], int *media){
    int i;
    for(i=0;i<N;i++)
    {
        *media = *media + array[i];
    }

    *media = *media/N;
}
```

Cosa sarebbe cambiato se fosse stato...

```
int calcolaMedia(int array[])
```

La mediana

- Per calcolare la mediana di dati:
  - si ordinano gli n di dati in ordine crescente o decrescente;
  - se il numero di dati è dispari la mediana corrisponde al valore centrale, ovvero al valore che occupa la posizione $(n + 1) / 2$
  - se il numero n di dati è pari, la mediana è stimata utilizzando i due valori che occupano le posizione $(n / 2)$ e $((n /2) + 1)$ (generalmente si sceglie la loro media

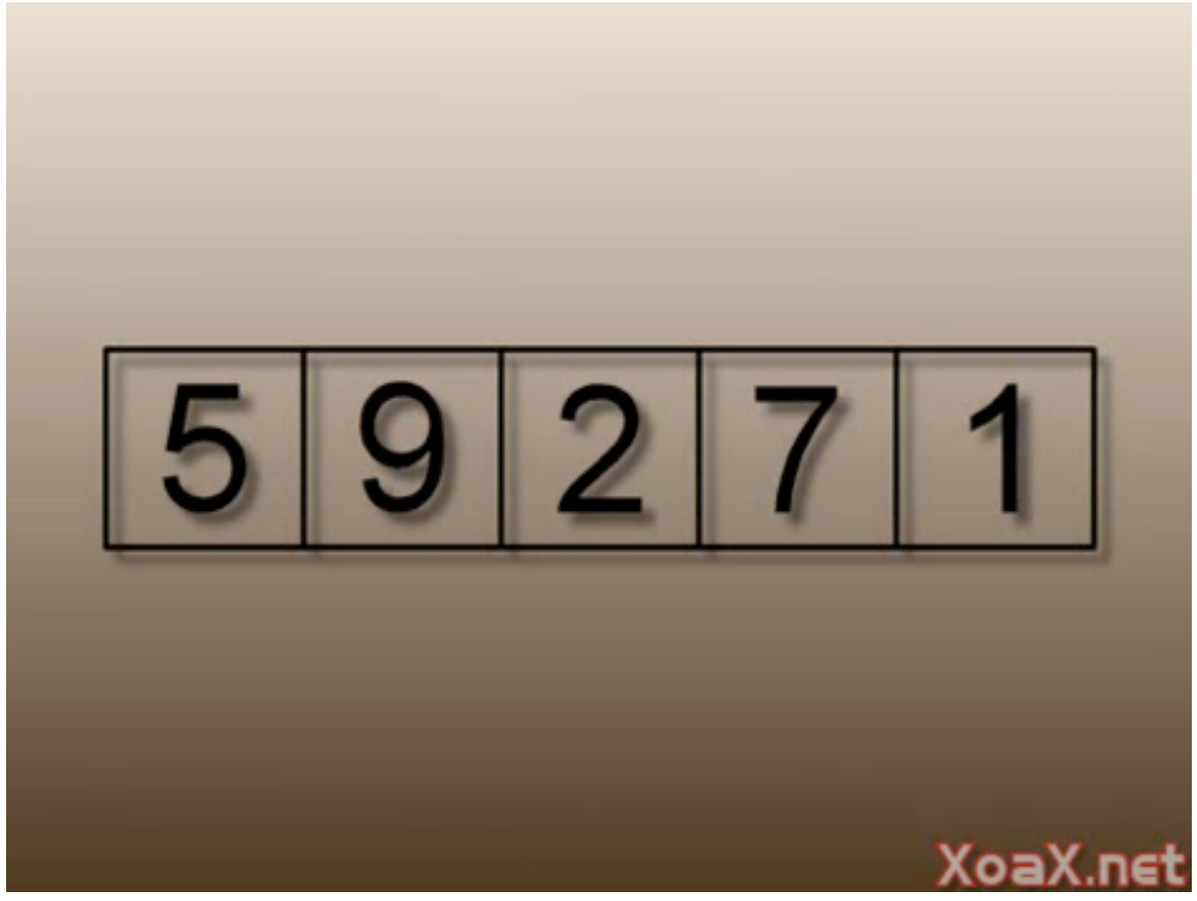
La mediana

- Per calcolare la mediana di dati:
  - si ordinano gli n di dati in ordine crescente o decrescente;

    Come facciamo ad ordinare i dati?
  - se il numero di dati è dispari la mediana corrisponde al valore centrale, ovvero al valore che occupa la posizione $(n + 1) / 2$
  - se il numero n di dati è pari, la mediana è stimata utilizzando i due valori che occupano le posizione $(n / 2)$ e $((n /2) + 1)$ (generalmente si sceglie la loro media

Ordinamento dei dati: bubble sort

- Per avere un ordinamento crescente:
  - scambio man mano a due a due due elementi x_i e x_{i+1} se $x_i > x_{i+1}$
- Dopo n-1 iteraizoni ho la garanzia di aver ordinato l'intero array



Ordinamento dei dati: bubble sort

- Per avere un ordinamento crescente:
  - scambio man mano a due a due due elementi x_i e x_{i+1} se $x_i > x_{i+1}$
- Dopo n-1 iteraizoni ho la garanzia di aver ordinato l'intero array

```
void bubbleSort(int arr[]){

    int i,j;

    for ( i = 1 ; i < N ; i++ )
        for ( j = 0 ; j < N - i ; j++ )
        {
                if (arr[j] > arr[j + 1])
                    swap(&arr[j], &arr[j+1]);


        }

}
```

Ordinamento dei dati: bubble sort

- Per avere un ordinamento crescente:
  - scambio man mano a due a due due elementi x_i e x_{i+1} se $x_i > x_{i+1}$
- Dopo n-1 iteraizoni ho la garanzia di aver ordinato l'intero array

```
void bubbleSort(int arr[]){

    int i,j;

    for ( i = 1 ; i < N ; i++ )
        for ( j = 0 ; j < N - i ; j++ )
        {
                if (arr[j] > arr[j + 1])
                    swap(&arr[j], &arr[j+1]);

        }

}
```

Ordinamento dei dati: bubble sort

- Per avere un ordinamento crescente:
  - scambio man mano a due a due due elementi x_i e x_{i+1} se $x_i > x_{i+1}$
- Dopo n-1 iteraizoni ho la garanzia di aver ordinato l'intero array

```
void swap(int *a, int *b)
{
     int tmp;
     tmp = *a;
     *a = *b;
     *b = tmp;
}
```

```
void bubbleSort(int arr[]){

     int i,j;

     for ( i = 1 ; i < N ; i++ )
          for ( j = 0 ; j < N - i ; j++ )
          {
                    if (arr[j] > arr[j + 1])
                      swap(&arr[j], &arr[j+1]);


          }

}
```

La mediana (codice C)

```c
//Funzione che calcola la mediana di un array
void calcolaMediana(int array[], int *mediana){

}
```

La mediana (codice C)

```c
//Funzione che calcola la mediana di un array
void calcolaMediana(int array[], int *mediana){

    //Ordino l'array in ordine crescente
    bubbleSort(array);

    //Distinguo i due casi in cui l'array ha numero di elementi pari o dispari
    if(N%2!=0)
        *mediana = array[N/2+1];
    else
        *mediana = (array[N/2]+array[N/2+1])/2;

}
```

Calcolo delle frequenze

```
//Funzione che calcola le frequenze dei valori dei lanci
void frequenze(int array[], int freq[]){

}
```

Calcolo delle frequenze

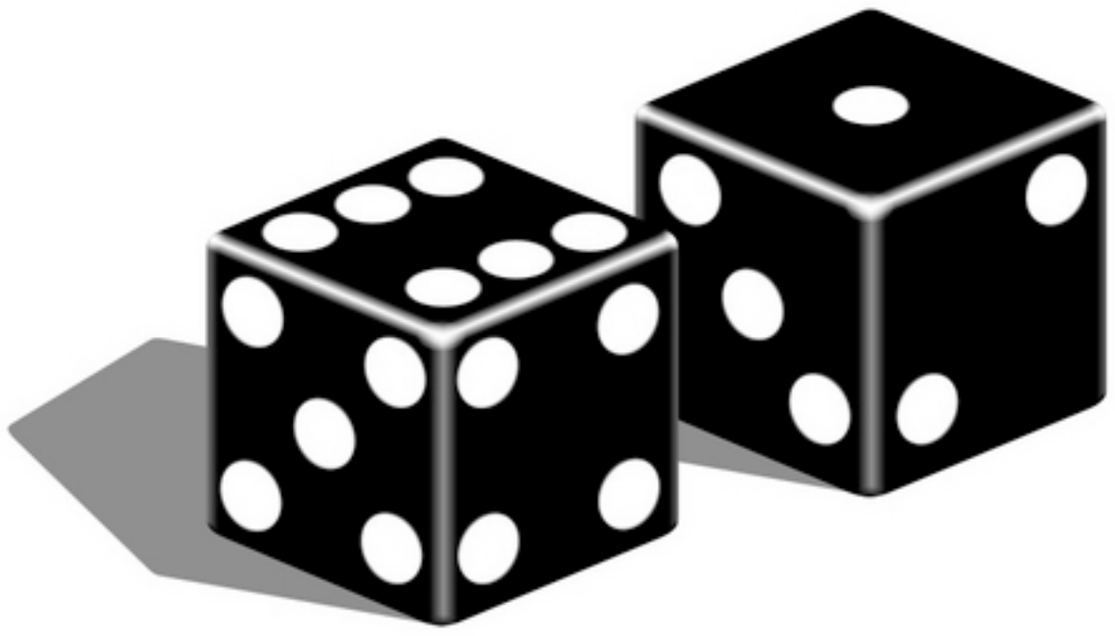
```
//Funzione che calcola le frequenze dei valori dei lanci
void frequenze(int array[], int freq[]){

    //Aggiungo +1 alla frequenza corrispondente
    //all'elemento dell'array -1
    //Esempio: array[i]= 5, aggiungo +1 a freq[array[i]-1]
    //cioe' freq[4] che rappresenta la frequenza di 5


    int i;

    for(i=0;i<N;i++)
        freq[array[i]-1]++;


}
```

Il main()

```c
int main(){

    int lanci[N]={3,4,6,1,3,5,4,2,6,2,3};

    //array contenente le frequenze dei valori da 0 a 5, rappresentanti i valori da 1 a 6 del dado
    int frequenze_array[6]={};
    int i;


    //Stampa dei lanci
    for(i=0;i<N;i++)
    {
        printf("%d ", lanci[i]);
    }

    int media=0, mediana;
    calcolaMedia(lanci, &media);
    calcolaMediana(lanci, &mediana);
    frequenze(lanci, frequenze_array);

    printf("\nLa media e' %d\n", media);
    printf("La mediana e' %d\n", mediana);
    for(i=0;i<6;i++)
    {
        printf("La frequenza di %d e' %d\n", i+1, frequenze_array[i]);
    }

    return 0;

}
```

FINE ESERCIZIO

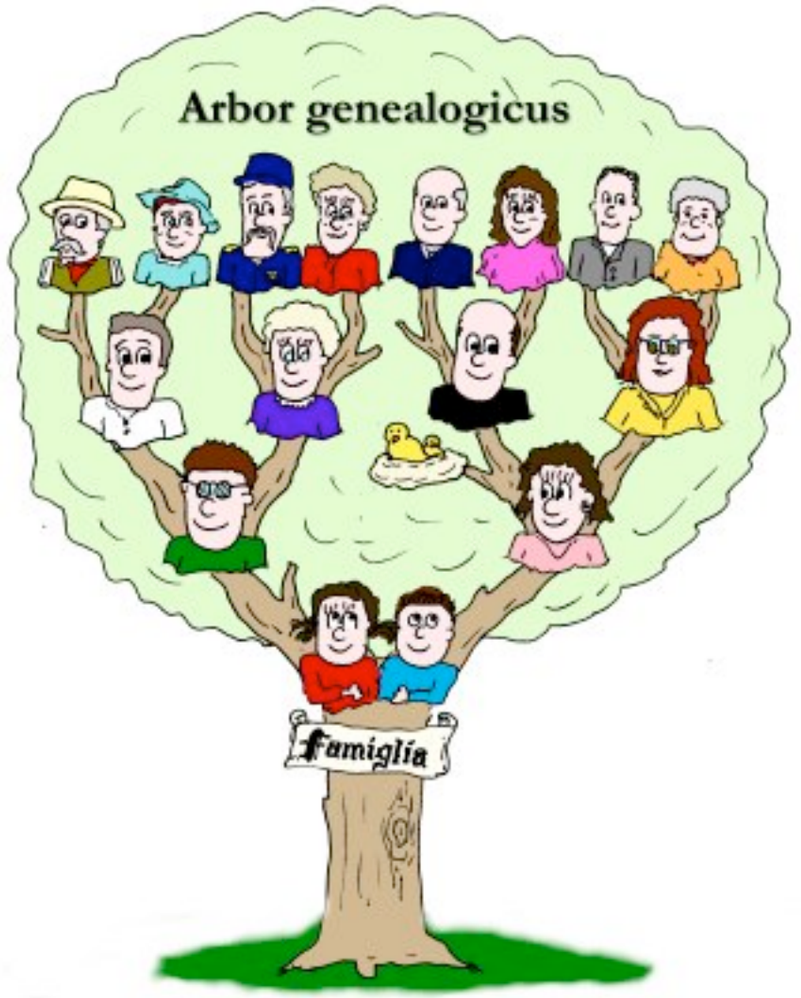
Domande?



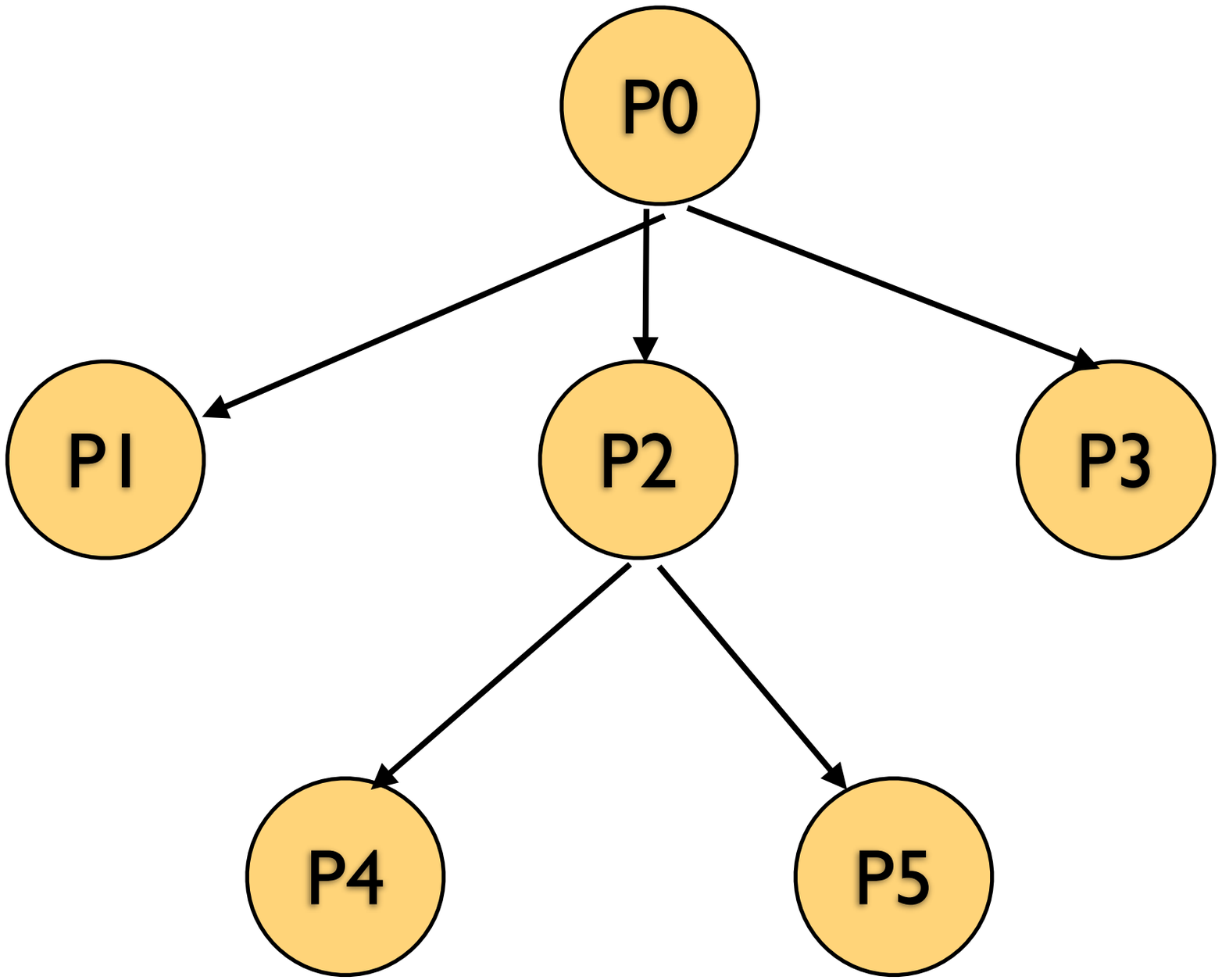
L'albero genealogico

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

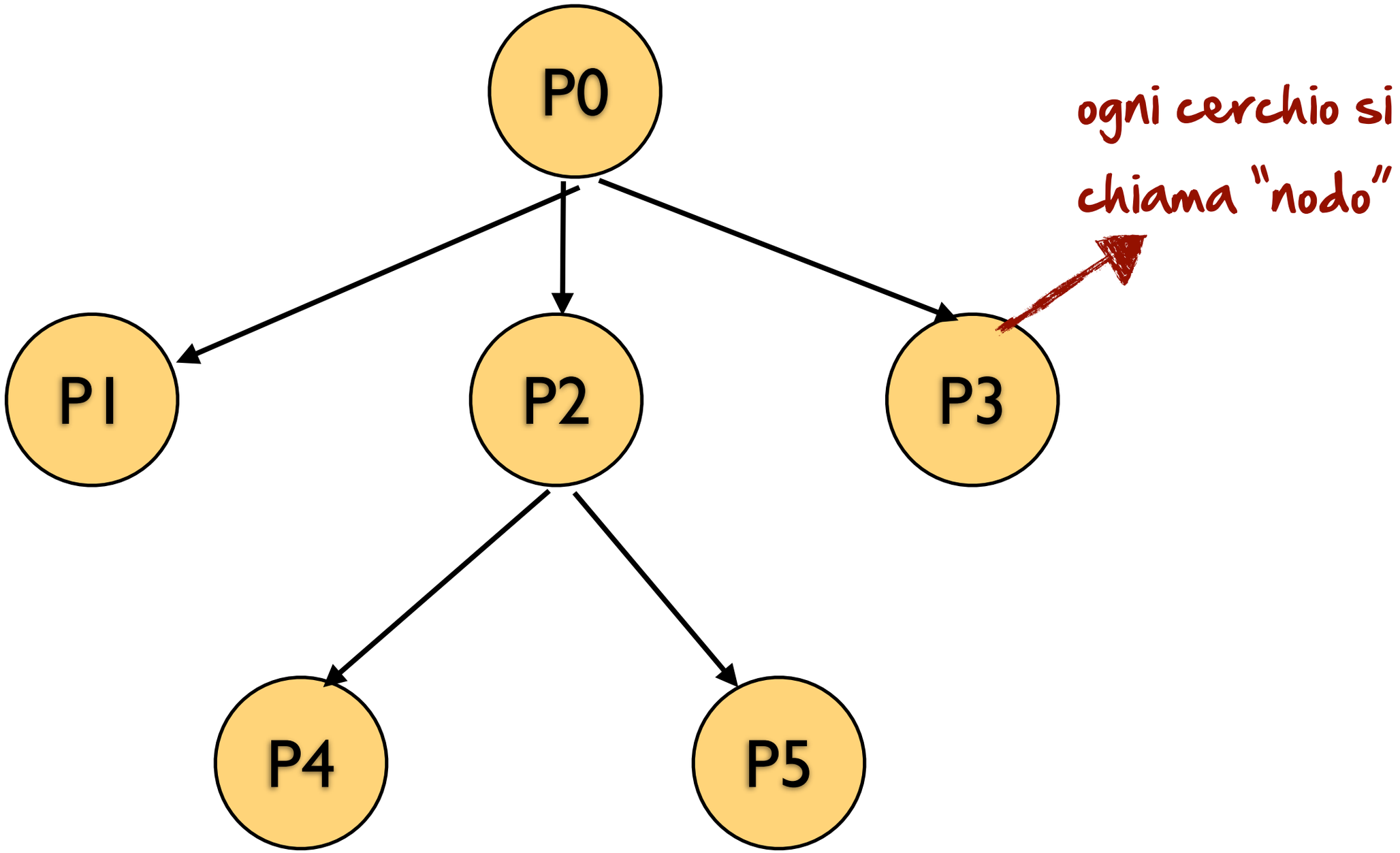
- In particolare, vogliamo poter fare:
  - Creare una persona
  - Rappresentare di una popolazione
  - Aggiungere figli ad una persona
  - Elencare i figli e i nipoti dato un antenato



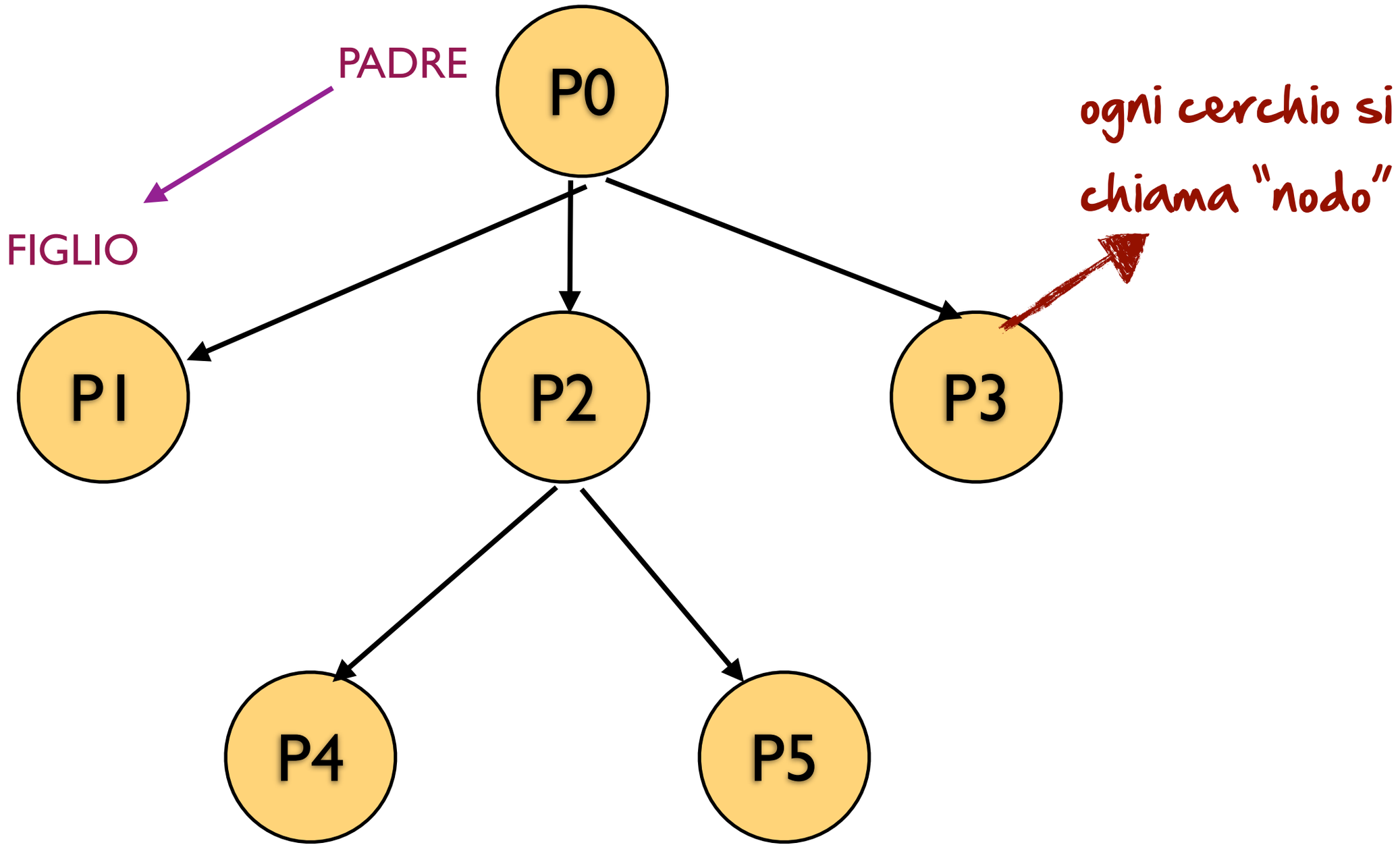
Una famosa struttura dati: l'albero



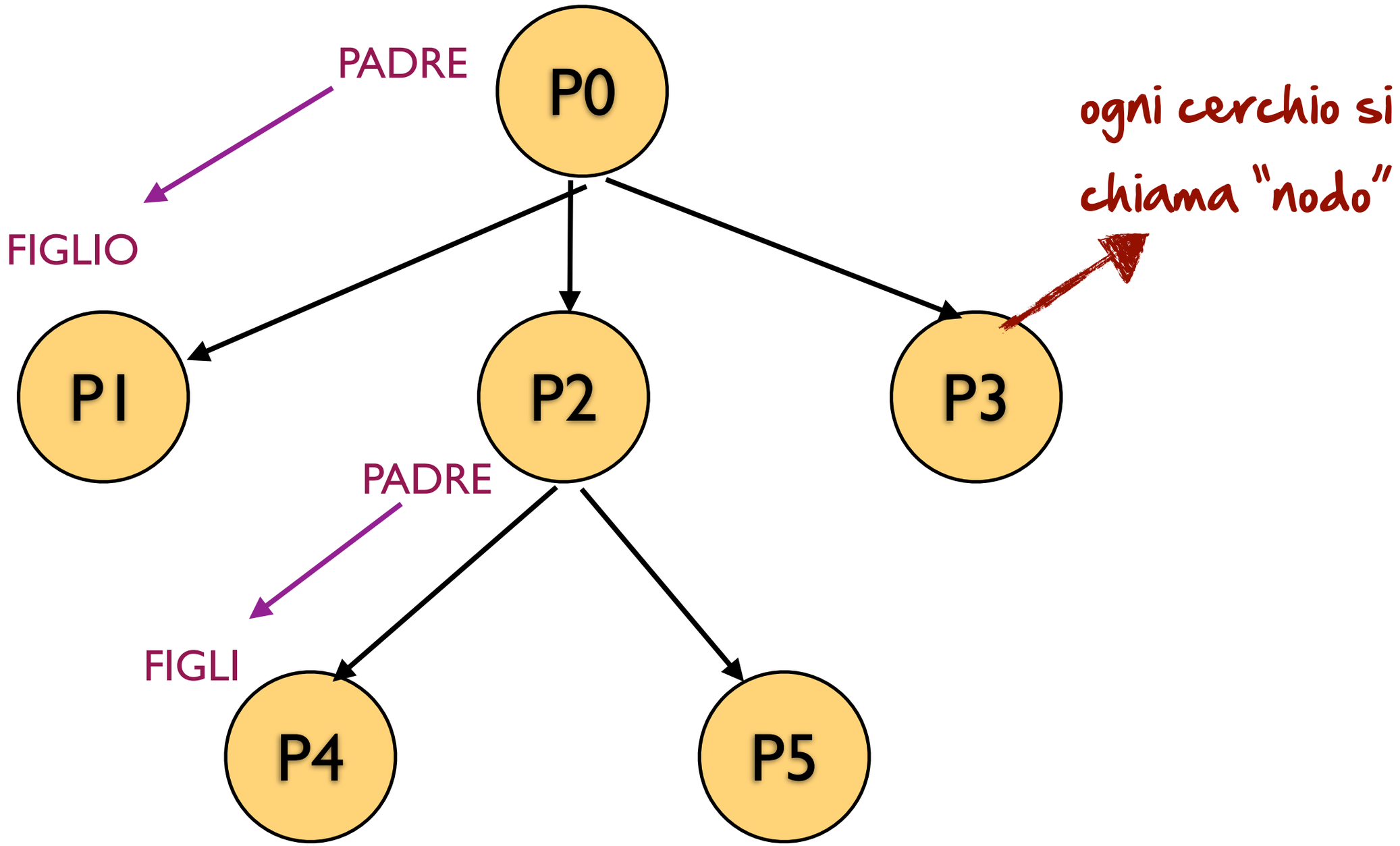
Una famosa struttura dati: l'albero



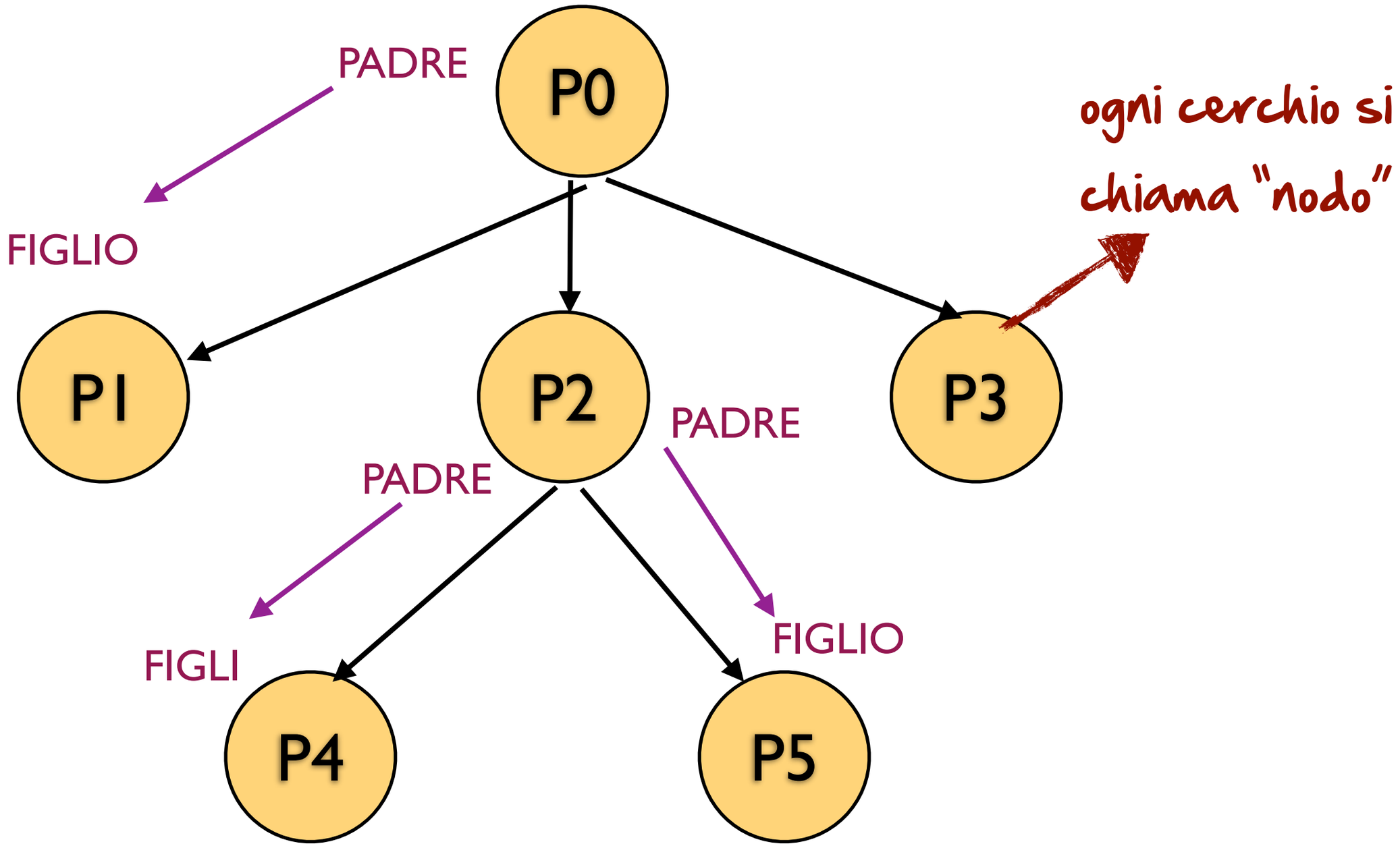
Una famosa struttura dati: l'albero



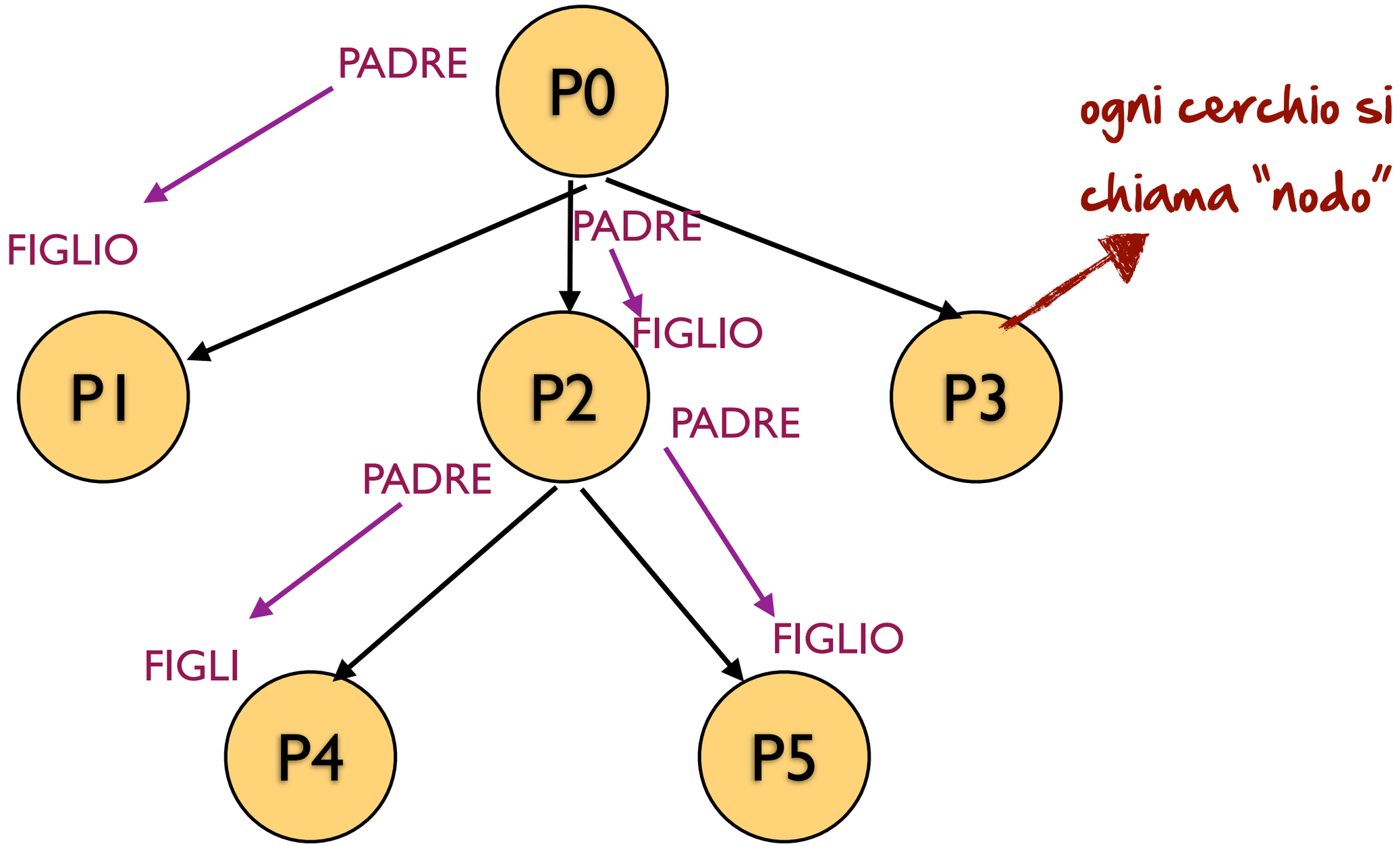
Una famosa struttura dati: l'albero



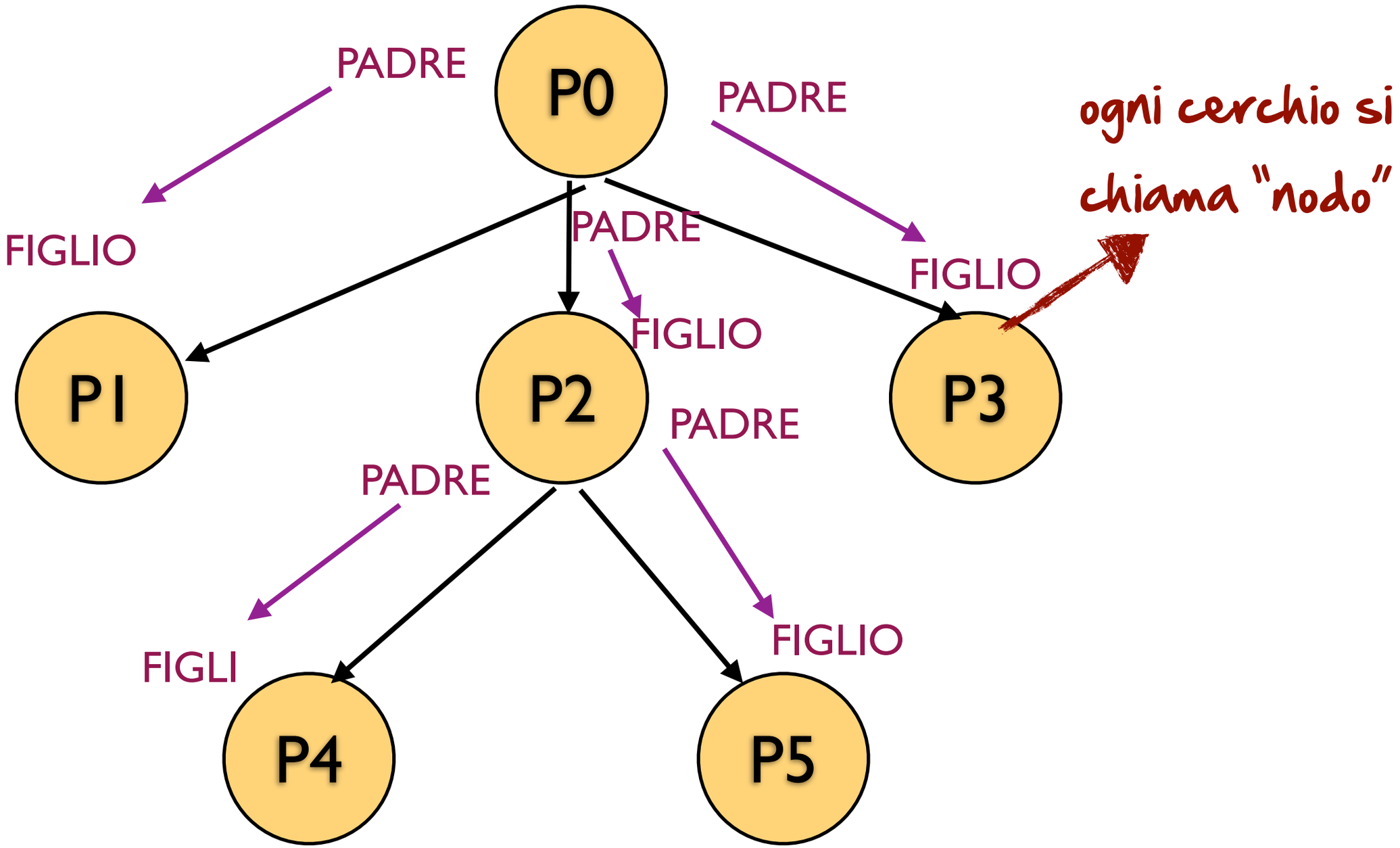
Una famosa struttura dati: l'albero



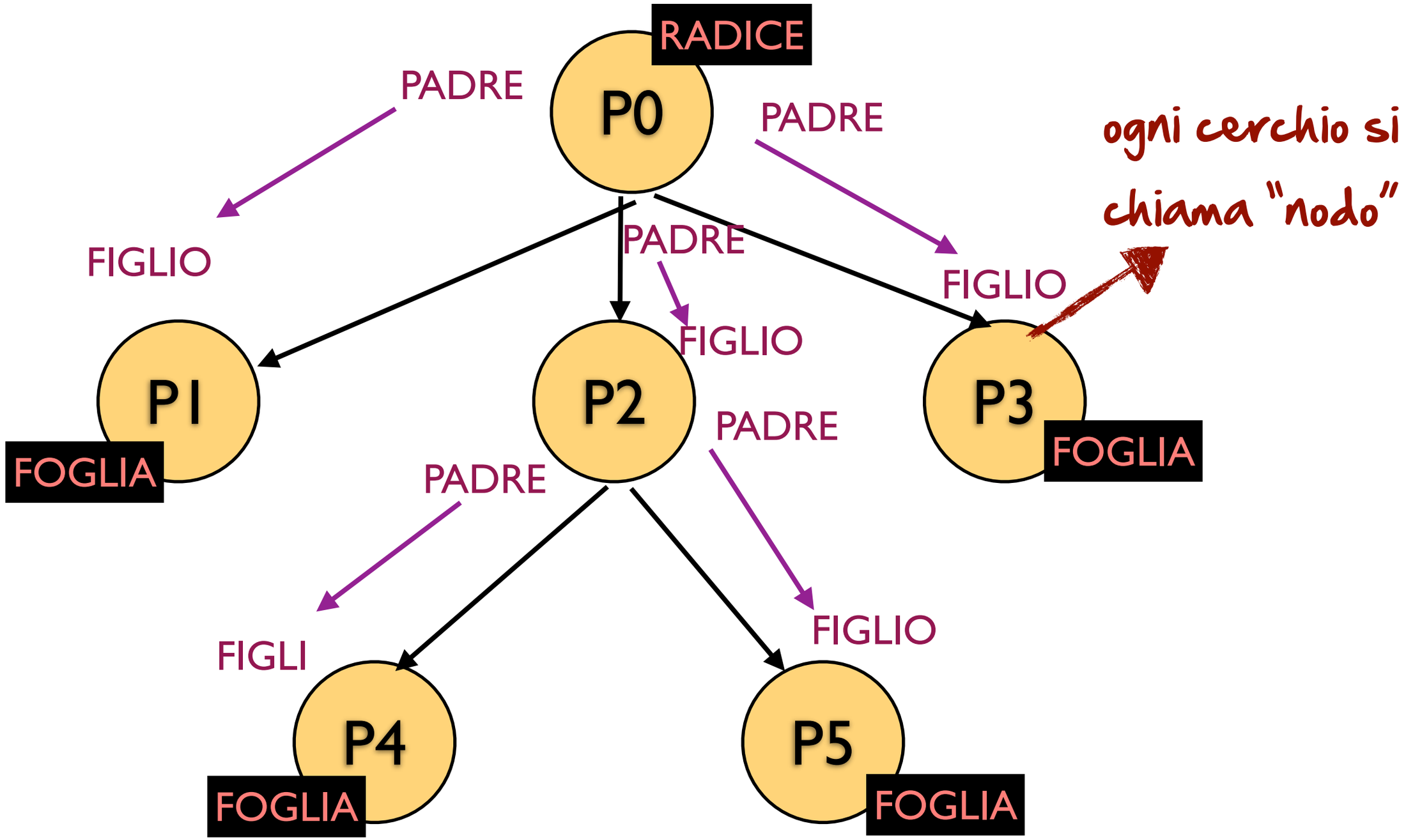
Una famosa struttura dati: l'albero



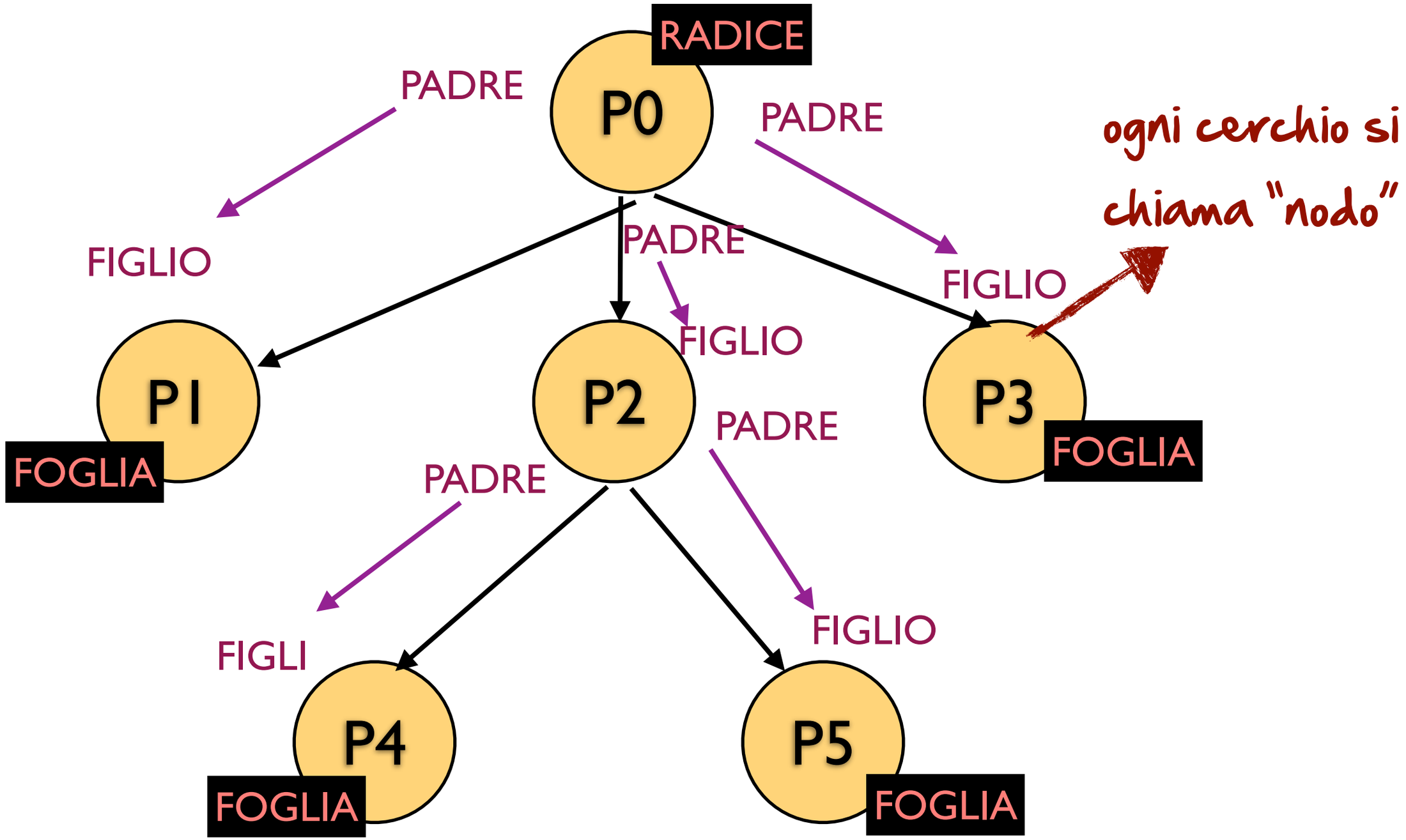
Una famosa struttura dati: l'albero



Una famosa struttura dati: l'albero

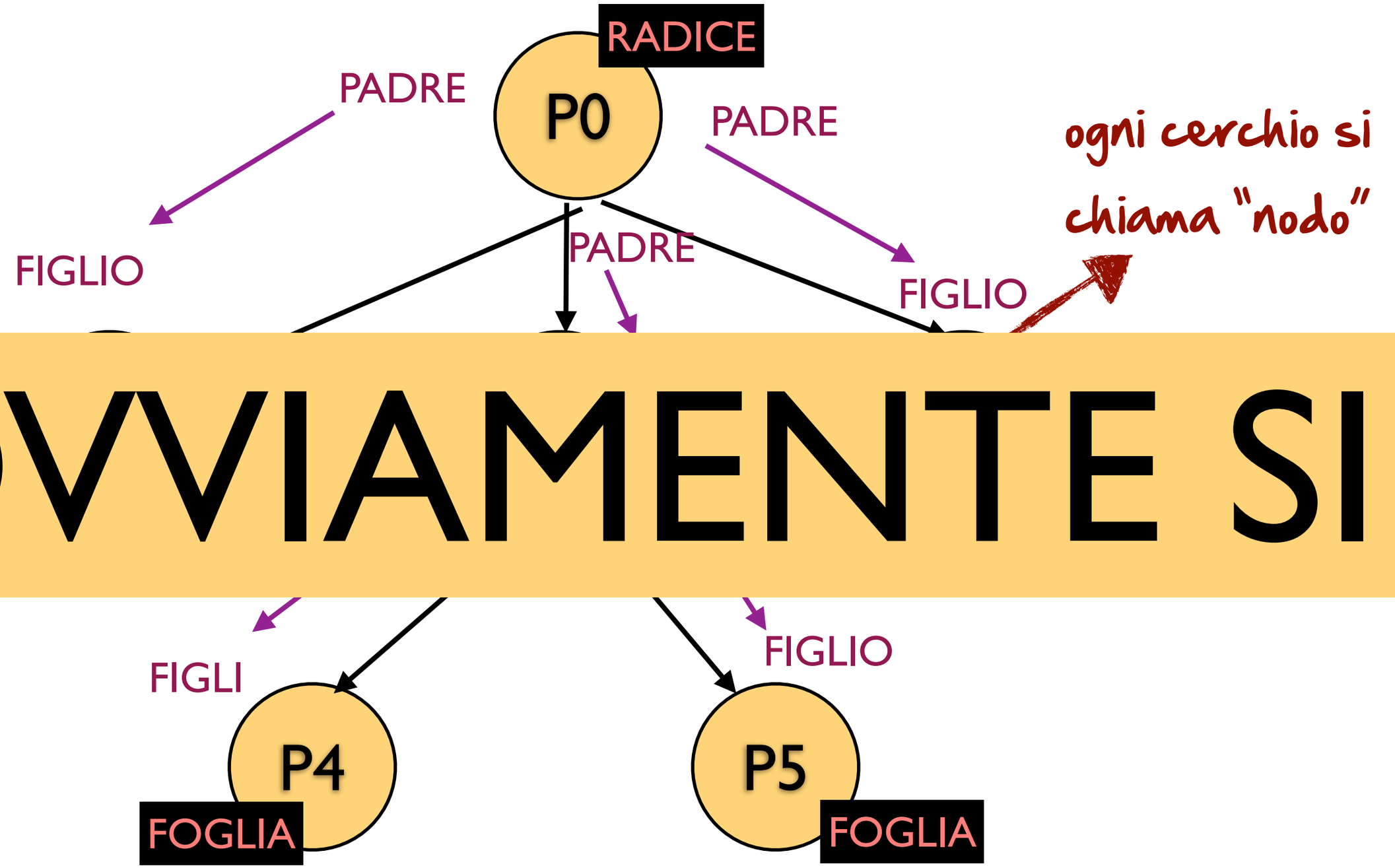


Una famosa struttura dati: l'albero



Può essere utile per rappresentare un albero genealogico?

Una famosa struttura dati: l'albero

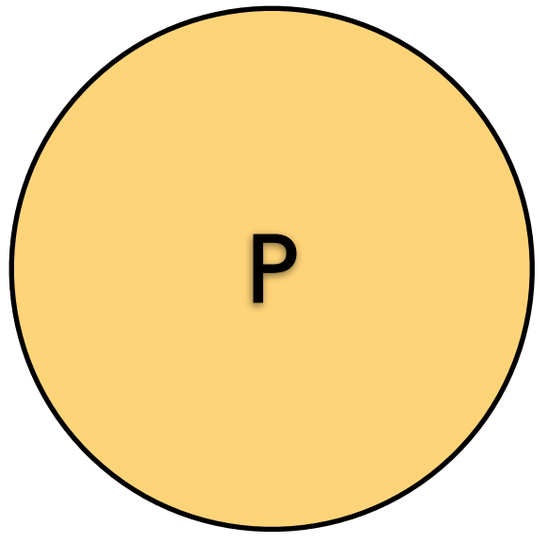


OVVIAMENTE SI

Può essere utile per rappresentare un albero genealogico?

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

==

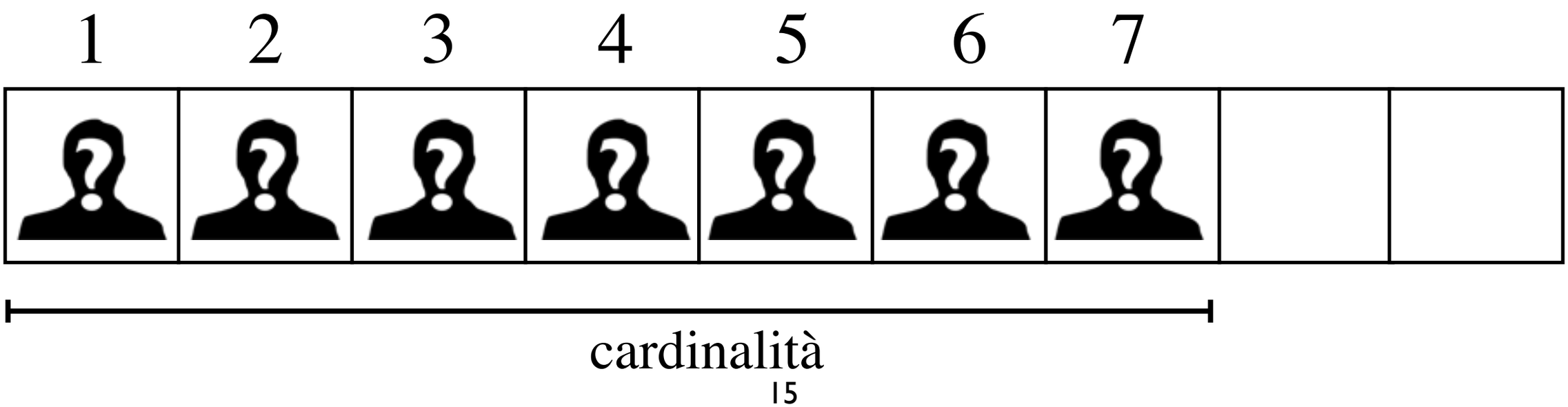


Una Persona

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

Una Popolazione

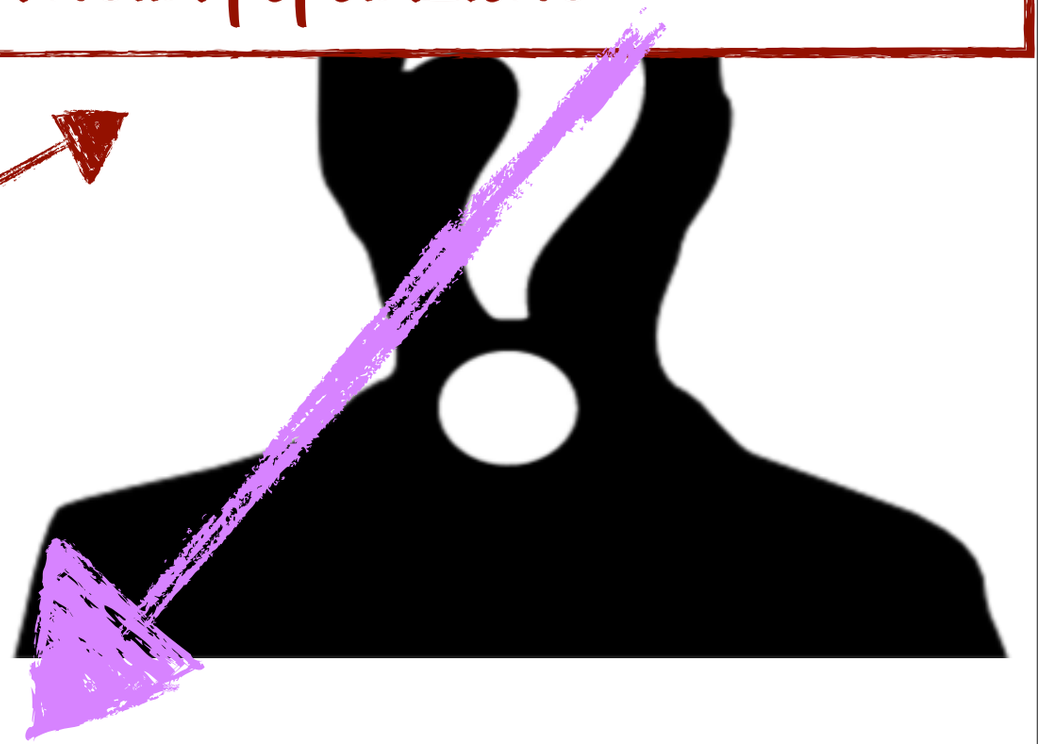
- Una popolazione è rapprentata da un insieme di persone
- Ogni persona ha un suo indice
- Esiste un numero di persone della popolazione (cardinalità)

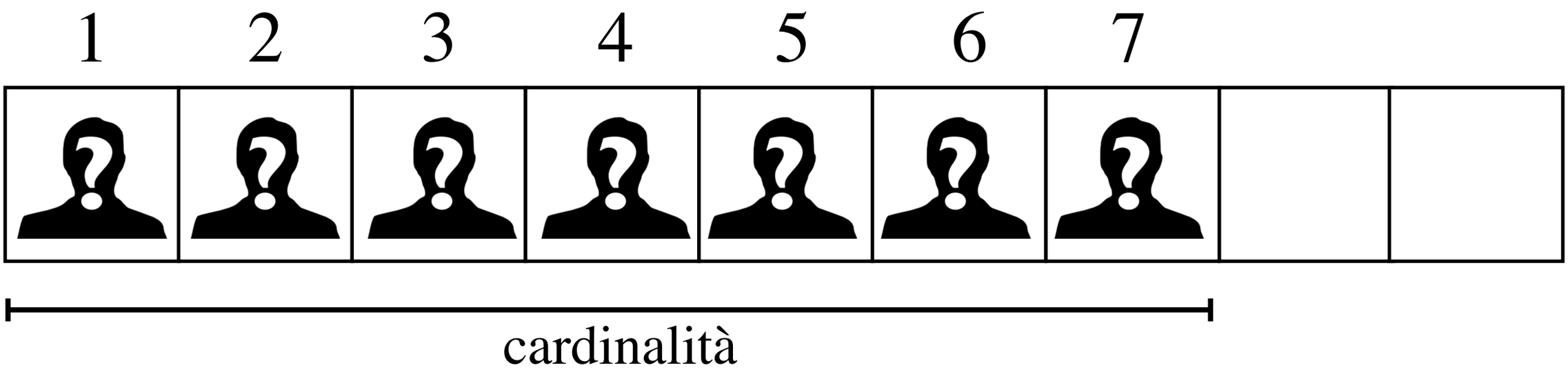


Una Persona nella popolazione

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

Li rappresentiamo con l'indice della persona nella popolazione





Persona e Popolazione (codice C)

```c
typedef struct {

} persona;
```

```c
int main () {

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

Persona e Popolazione (codice C)

```c
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;
```

Persona e Popolazione (codice C)

```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

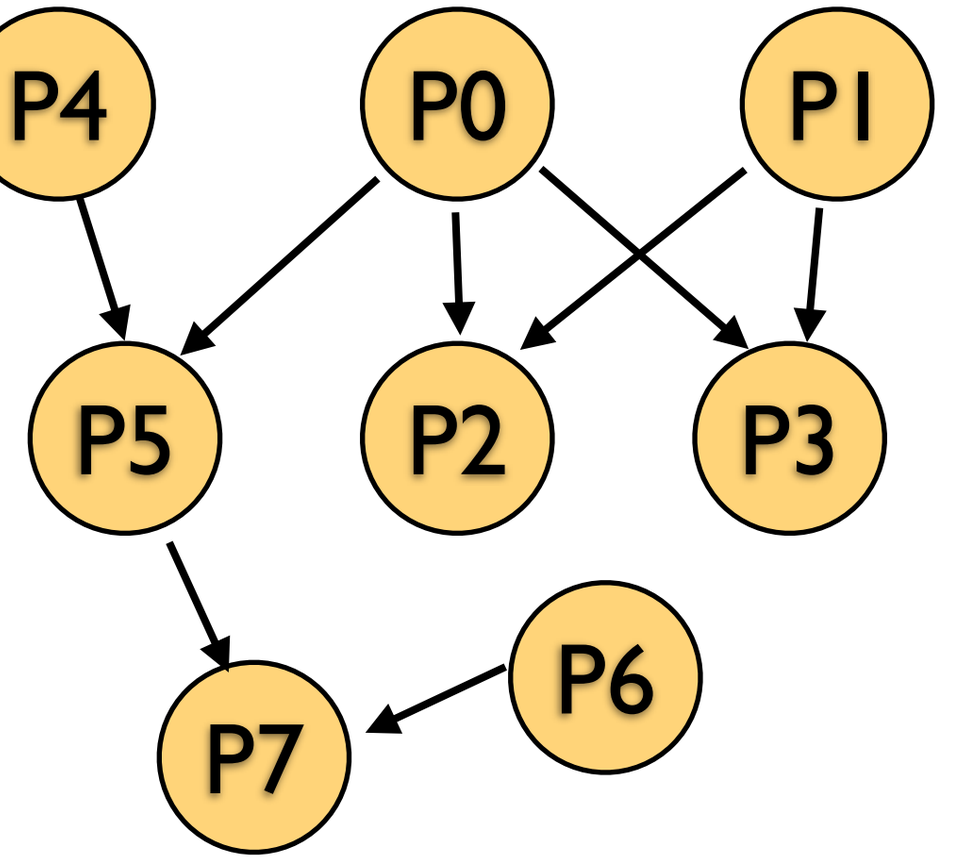
MARCO	STEFANIA	LUCA	PIPPO	LUCIA	ARIANNA	RINALDO	STEFANO
P0	P1	P2	P3	P4	P5	P6	P7

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

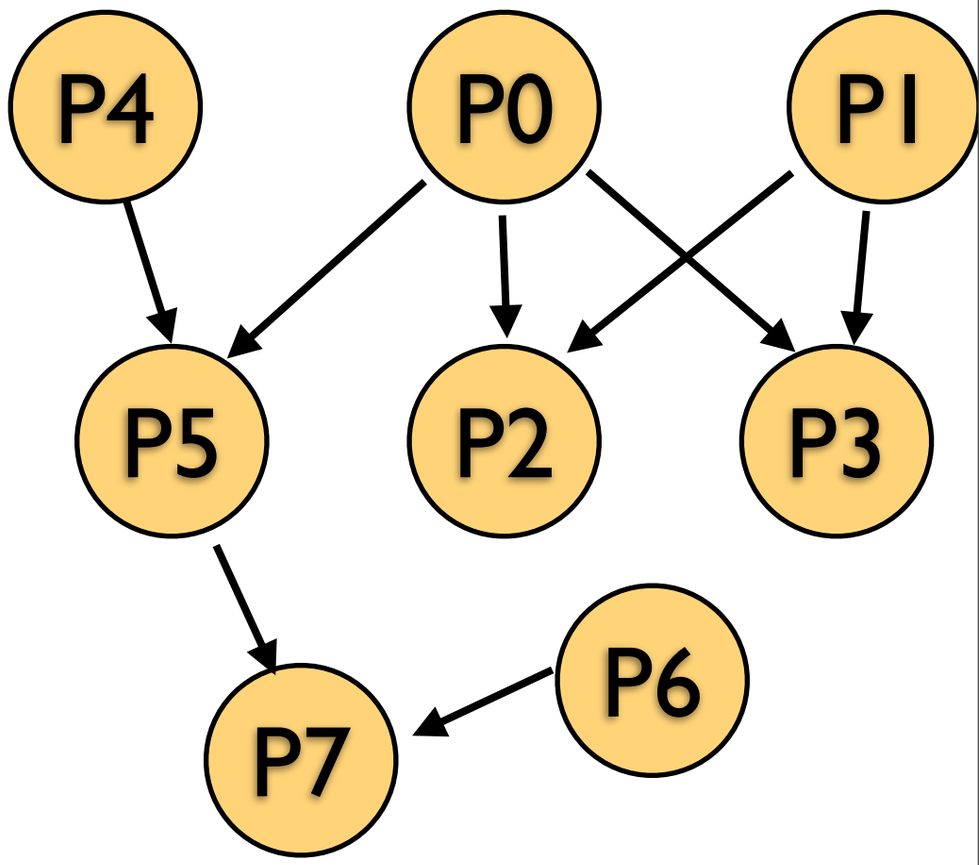
Marco e' padre di LUCA e di PIPPO
Stefania e' madre di LUCA e di PIPPO
Arianna e' figlia di Marco e Lucia
Stefano e' figlio di Arianna e Rinaldo



La nostra popolazione (codice C)

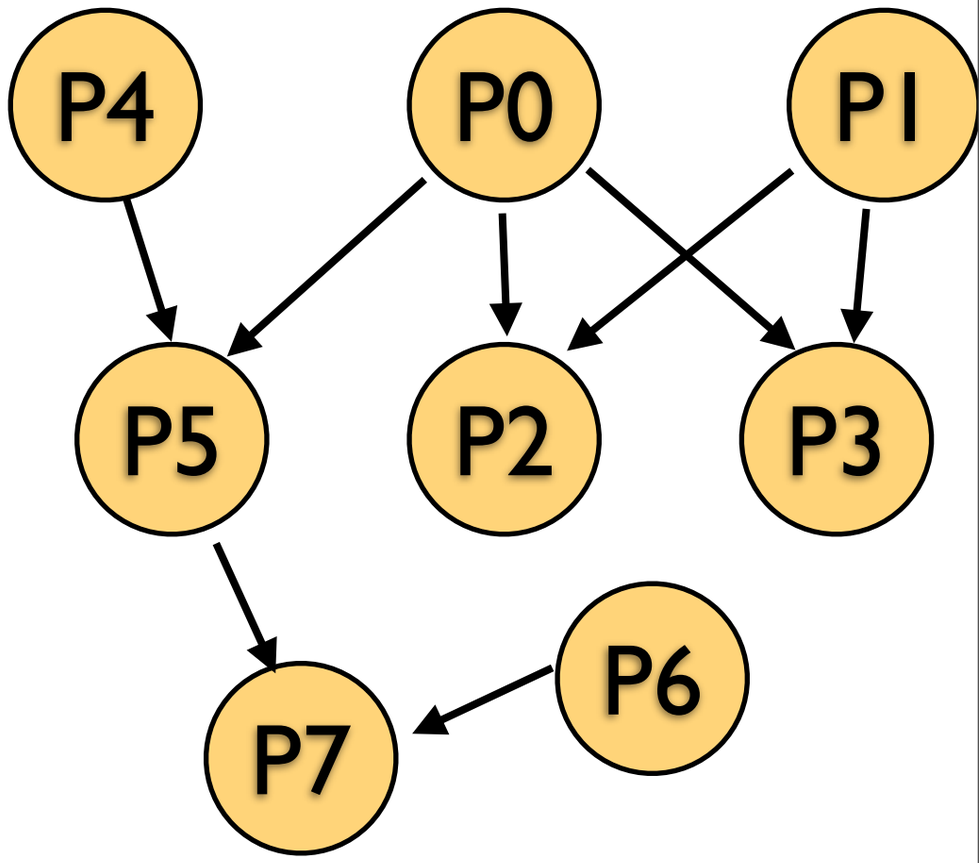
```c
int main ()
{

    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;

}
```

Potete lasciare il vostro giudizio qui:

http://tinyurl.com/lElMExe2013

Tutte il materiale sarà disponibile sul mio sito internet:

alessandronacci.com