**5. RICORSIONE**

Dato un oggetto, come funzioni, insiemi, algoritmi, … in alcuni casi esso può essere ***definito in termini di sé stesso, ma di più piccole dimensioni***.

*Esempi*:

|  |  |  |
| --- | --- | --- |
|  |  |  |

**5.1 DEFINIRE FUNZINI RICORSIVE**

|  |  |
| --- | --- |
| In alcuni casi la definizione ricorsiva di un oggetto può essere molto facile da scrivere. | In altri casi la definizione ricorsiva di un oggetto è l’unico modo per descriverlo. |

Per definire una funzione ricorsiva ***sull’insieme degli interi non negativi***.

1. (***Passo base***) Specificare il valore della funzione in 0
2. (***Passo ricorsivo***) Dare una regola per determinare il valore della funzione in n in termini del valore della funzione in interi n-1

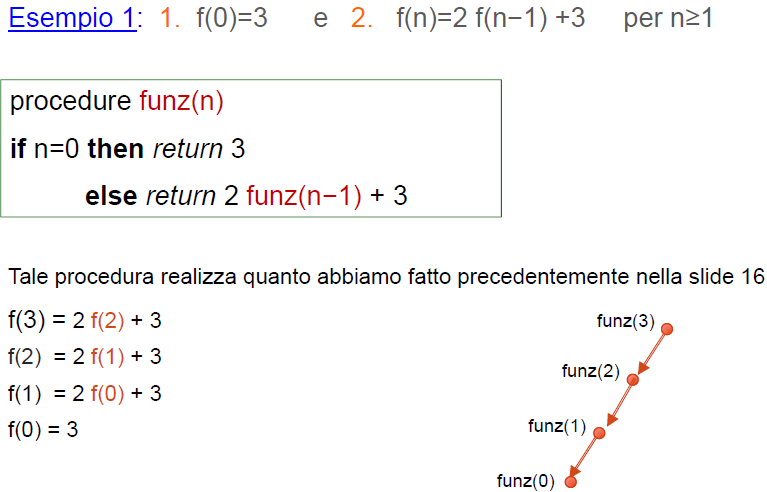
|  |  |  |
| --- | --- | --- |
|  |  |  |

**5.2 CALCOLO DI FUNZIONI RICORSIVE**

|  |  |  |
| --- | --- | --- |
|  |  |  |

**5.3 USO DELLA RICORSIONE**

Un algoritmo è detto ***ricorsivo*** se risolve un problema riducendo esso ad una istanza dello stesso problema ma con un input più piccolo.

**Correttezza degli Algoritmi Ricorsivi**: (correttezza = produce output corretto per ogni possibile input)

Proviamo la correttezza dell’algoritmo descritto dalla procedura ***funz(n)***

Dimostriamo, cioè che il valore restituito dalla procedura ***funz(n)*** coincide con ***f(n)***

**Dimostrazione**:

Usiamo l’induzione matematica su n.

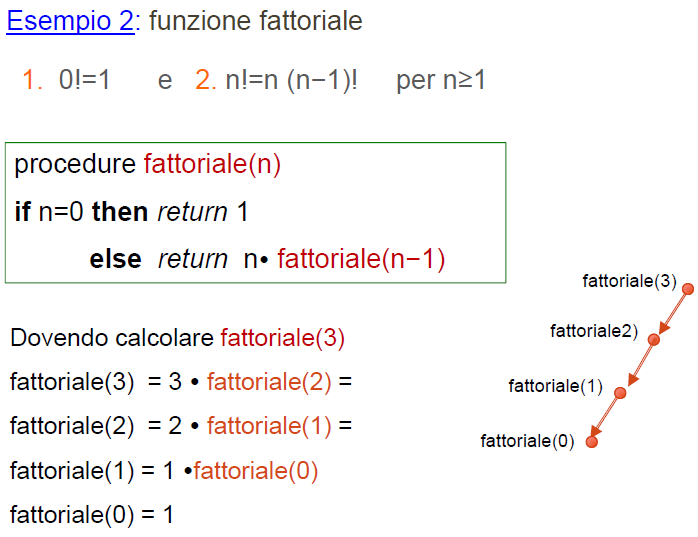
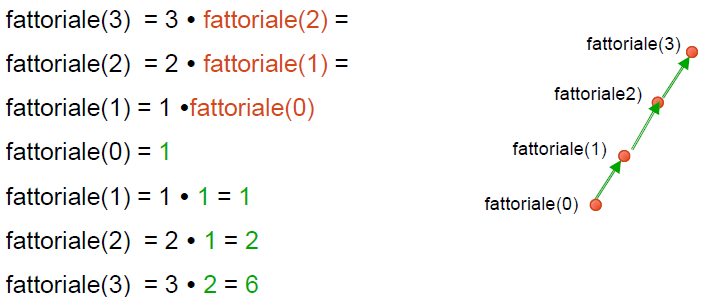
***Base***: Se n=0, il primo passo dell’algoritmo ci dice che il valore restituito da ***funz(0)*** è 3. Corretto perché ***f(0)***=3.

***Ipotesi induttiva***: per un n intero positivo arbitrario, l’algoritmo computa correttamente f(n), cioè ***funz(n)*** restituisce ***f(n)***.

***Passo di induzione***: Ora mostriamo che la procedura ***funz(n+1)*** computa correttamente anche ***f(n+1).***

La procedura ***funz(n+1)*** restituisce 2 funz(n) + 3.

Per ipotesi induttiva funz(n) coincide con f(n), quindi 2 funz(n) + 3 coincide con 2 f(n) +3 = ***f(n+1)***

Proviamo la correttezza dell’algoritmo descritto dalla procedura ***fattoriale(n)***

Dimostriamo, cioè che il valore restituito dalla procedura ***fattoriale(n)*** coincide con ***n!***

**Dimostrazione**:

Usiamo l’induzione matematica su n.

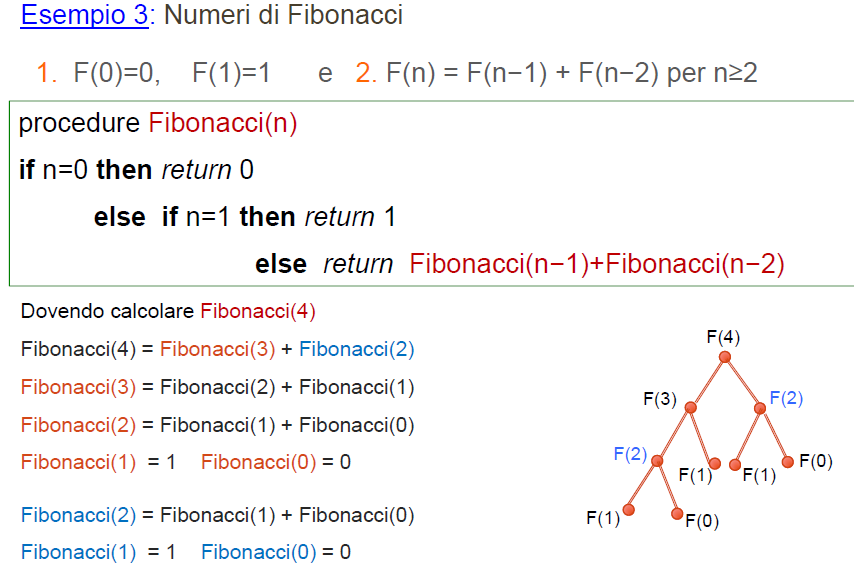
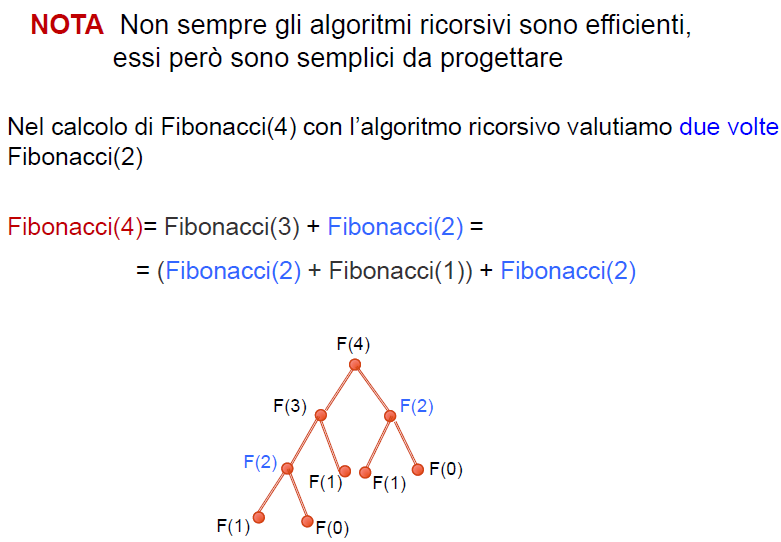
***Base***: Se n=0, il primo passo dell’algoritmo ci dice che il valore restituito da ***fattoriale(0)*** è 1. Corretto perché ***0!***=1.

***Ipotesi induttiva***: per un n intero positivo arbitrario, l’algoritmo computa correttamente n!, cioè ***fattoriale(n)*** restituisce ***n!***

***Passo di induzione***: Ora mostriamo che la procedura ***fattoriale(n+1)*** computa correttamente anche ***(n+1)!***

La procedura ***fattoriale(n+1)*** restituisce (n+1) \* fattoriale(n)

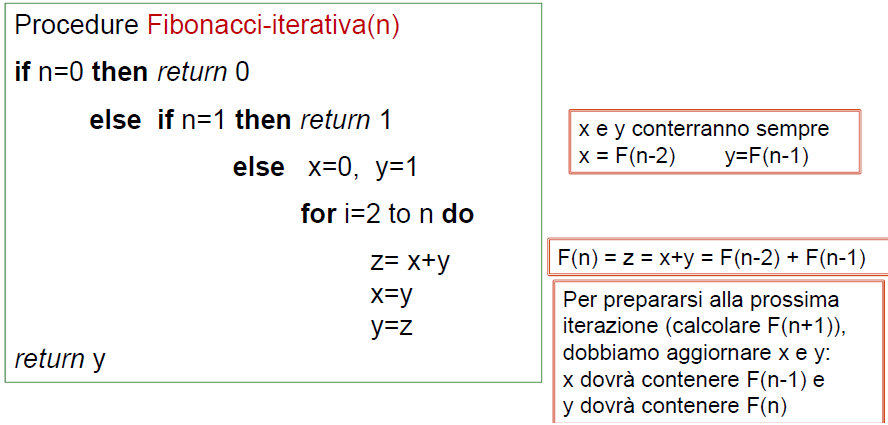
Per ipotesi induttiva, (n+1) \* fattoriale(n) coincide con (n+1)\*n!= ***(n+1)!***

Consideriamo ora una ***procedura iterativa*** per il calcolo dei ***numeri di Fibonacci***:

1. F(0)=0, F(1)=1

2. F(n) = F(n-1) + F(n-2) per n≥3

 Ciascun numero di Fibonacci viene calcolato esattamente ***una volta***.

**5.4 USO DI DEFINIZIONI RICORSIVE**

Le definizioni ricorsive possono essere usate nelle ***dimostrazioni***:

*Esempio*:

Sia F(n) n-simo termine della sequenza dei numeri di Fibonacci.

Provare che per n≥3, F(n) > αn−2 dove α= (1+√5)/2

***Dimostrazione***:

Usiamo l’induzione forte:

***Base***: F(3) = F(2) + F(1) = 1+1 = 2 > (1+√5)/2 = α

F(4) = F(3) + F(2) = 2+1 = 3 > [(1+√5)/2]2 = α2

***Ipotesi induttiva***: Assumiamo che per 3≤j≤n con n≥3

si ha F(j) > αj−2

***Passo di induzione***: Consideriamo ora F(n+1) e la sua definizione, poi applichiamo l’ipotesi induttiva

F(n+1) = F(n−1)+F(n) > αn−3 + αn−2 = αn−3 (1+α) =αn−3 α2 =αn−1

***Ricordate***: Un ***insieme*** può essere definito:

|  |  |
| --- | --- |
| Elencando i suoi elementi***: {a, b, c} ha elementi a,b,c*** | Specificando le proprietà caratteristiche di suoi elementi ***A = {w | w ha la proprieta P}*** |

Un altro modo per descrivere ***insiemi*** è attraverso una definizione ricorsiva:

|  |
| --- |
| Un insieme A è definito ricorsivamente nel modo seguente:  ***Passo base***: Si definiscono uno o più oggetti elementari  ***Passo ricorsivo***: definisce la regola che permette di costruire oggetti più complessi in termini di quelli già definiti dell'insieme. |
| **Nota**: Gli elementi dell’insieme sono definiti esclusivamente dalle regole date. |

*Esempio*:

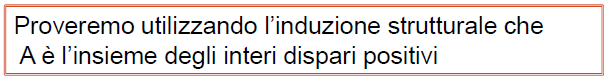
Sia A un sottoinsieme di interi definito ricorsivamente come segue:

***Passo base***: 1 ∈ A

***Passo*** ***ricorsivo***: se x ∈ A allora x + 2 ∈ A

Quali sono gli elementi di A?

1 ∈ A ***Passo base***

Applico il ***Passo ricorsivo***: x=1 ∈ A allora x + 2 = 1 + 2 = 3 ∈ A

Applico il ***Passo ricorsivo***: x=3 ∈ A allora x + 2 = 3 + 2 = 5 ∈ A

1, 3, 5, 7, 9 … ∈ A

|  |  |
| --- | --- |
| Le definizioni ricorsive possono essere usate per descrivere ***insiemi di stringhe***.  Un ***alfabeto*** e un insieme finito di elementi (chiamati lettere o simboli): | L'alfabeto delle ***lettere romane minuscole*** è Σ= {a,b,…,z}.  L'alfabeto delle ***cifre arabe*** è Σ= {0,1,….,9}.  L'alfabeto ***binario*** è Σ= {0,1}. |
| L’insieme di stringhe Σ\* sull’alfabeto Σ è definito ricorsivamente nel modo seguente:  ***Passo base***: la stringa vuota λ∈ Σ\*  ***Passo ricorsivo***: Se w ∈Σ\* e x ∈Σ allora wx ∈Σ\* | |

*Esempio*:

Sia Σ= {0,1}. Σ\* è l’insieme di tutte le stringhe binarie.

Infatti, λ∈ Σ\* applicando il ***Passo base***

0 e 1 ∈Σ\* applicando la prima volta il ***Passo ricorsivo***

00, 01, 10, 11 ∈Σ\* con la seconda applicazione

|  |  |
| --- | --- |
| Le definizioni ricorsive possono essere usate per ottenere ***nuove definizioni ricorsive:*** | La lunghezza ***l(w)*** di una parola ***w*** è definita come il numero di caratteri di cui ***w*** è costituita.  Σ= {a,b,…,z}, zaino ∈Σ\* l(zaino) = 5 |
| La ***lunghezza di una parola*** in Σ\* sull’alfabeto Σ è definito ricorsivamente nel modo seguente  ***Passo base***: l(λ) = 0  ***Passo ricorsivo***: Se ***w*** ∈Σ\* e x ∈Σ allora l(***w***x) = l(***w***) + 1 | |

|  |  |
| --- | --- |
| Le definizioni ricorsive possono essere usate per descrivere ***parole palindrome:*** | Una stringa è ***palindroma*** se letta da sinistra a destra o viceversa, è la stessa.  alla, otto, ingegni, Anna, ottetto. |
| L’insieme delle parole palindrome sull’alfabeto Σ={a, b} è definito ricorsivamente nel modo seguente:  ***Passo base***: a, b, λ sono parole palindrome  ***Passo ricorsivo***: Se w è una parola palindroma allora anche awa e bwb sono parole palindrome | |

*Esempio*:

abba è una parola palindroma:

* λ è una parola palindroma ***Passo base***
* bλb =bb è una parola palindroma ***Passo*** ***ricorsivo***
* a bb a è una parola palindroma ***Passo*** ***ricorsivo***

Le definizioni ricorsive possono essere usate per descrivere ***espressioni aritmetiche***.

|  |
| --- |
| Una ***espressione aritmetica*** è definita ricorsivamente nel modo seguente  ***Passo base***: i numeri (interi o reali) e le variabili sono espressioni aritmetiche  ***Passo ricorsivo***: se E1 ed E2 sono espressioni aritmetiche allora (E1 + E2), (E1 − E2), (E1 × E2), (E1\ E2) sono espressioni aritmetiche.  Se E è un'espressione aritmetica allora (−E) è un'espressione aritmetica. |

Le definizioni ricorsive possono essere usate per descrivere ***strutture dati***.

|  |  |
| --- | --- |
| Un ***albero radicato*** può essere descritto ricorsivamente nel modo seguente:  ***Base***: un singolo vertice r è un albero radicato  ***Passo*** ***ricorsivo***: Supponiamo che T1, T2, …, Tn sono alberi radicati disgiunti con radici r1, r2, …, rn.  Allora, il grafo formato dalla radice r, che non è in nessuno degli alberi radicati T1, T2, …, Tn,  ottenuto connettendo con un arco r a ciascun r1, r2, …, rn è anch’esso un albero radicato. |  |

Più formalmente, un albero radicato T=(V, E) è definito:

|  |
| --- |
| ***Base***: un singolo vertice r è un albero radicato, T = ({r}, ∅)  ***Passo*** ***Ricorsivo***: Supponiamo che T1= (V1,E1), T2= (V2,E2), …, Tn= (Vn,En) sono alberi radicati disgiunti, cioè Vi ∩ Vj = per 1 ≤ i≠j ≤ n con radici r1, r2, …, rn, dove r1 ∈ V1, r2 ∈ V2, …, rn ∈ Vn.  Allora, il grafo T=(V,E) con V={r} ∪ V1 ∪ V2 ∪ … ∪ Vn ed E = {(r,r1),(r,r2), …., (r,rn)} ∪ E1 ∪ E2 ∪ … ∪ En  formato dalla radice r che non è in nessuno degli alberi radicati T1, T2, …, Tn, dove r ∈ V e r ∉ V1 ∪ V2 ∪ … ∪ Vn ottenuto connettendo con un arco r a ciascun r1, r2, …, rn, dove (r,r1) ∈ E, (r,r2) ∈ E, …, (r,rn) ∈ E è anch’esso un albero radicato. |

Un ***albero binario pieno*** è un albero radicato dove ciascun vertice ha 0 o 2 figli; se tali figli esistono, essi sono chiamati figlio destro e figlio sinistro.

|  |  |
| --- | --- |
| Un **albero binario pieno** è descritto ricorsivamente nel modo seguente:  ***Base***: un singolo vertice r è un albero binario pieno.  ***Passo*** ***ricorsivo***: Se T1 e T2 sono alberi binari pieni, allora l’albero T formato connettendo la radice r con un arco alla radice del sottoalbero sinistro T1 e con un altro arco la radice del sottoalbero destro T2 è un albero binario pieno. |  |

Le definizioni ricorsive possono essere usate per ***ottenere nuove definizioni ricorsive***.

|  |  |
| --- | --- |
| **L’altezza h(T)** di un albero radicato T è definita ricorsivamente come:  ***Passo*** ***base***: h(T) = 0 se T consiste della sola radice r  ***Passo ricorsivo***: h(T) = 1 + max {h(T1), …, h(Tn)} se T1, …, Tn sono i sottoalberi di T |  |
| Il **numero di vertici n(T)** di un albero radicato T è definita ricorsivamente come:  ***Passo base***: n(T) = 1 se T consiste della sola radice r  ***Passo ricorsivo***: n(T) = 1 + n(T1) + … + n(Tn) se T1, …, Tn sono i sottoalberi di T |  |
| Il **numero di vertici con due figli d(T)** di un albero binario pieno T è definito ricorsivamente come:  ***Passo base***: d(T) = 0 se T consiste della sola radice  ***Passo ricorsivo***: d(T) = 1 + d(T1) + d(T2) se T1 e T2 sono i sottoalberi di T |  |
| Il **numero di foglie f(T)** di un albero binario pieno T è definito ricorsivamente come:  ***Passo base***: f(T) = 1 se T consiste della sola radice  ***Passo ricorsivo***: f(T) = f(T1) + f(T2) se T1 e T2 sono i sottoalberi di T |  |