
Programmazione 1

Esercitazione 3

Cognome:

Nome:

Matricola:

1. Mostrare come le due funzioni **Sommatoria** e **Produttoria** viste nell'esercitazione precedente, sono entrambe dei casi particolari di una più generica funzione che possiamo chiamare **Accumula**:

```
1 def Accumula(Operazione, ElementoNeutro, F, a, Next, b):  
2     # DA COMPLETARE COME ESERCIZIO
```

La funzione **Accumula** prende in input tre funzioni e tre valori:

- (a) **Operazione**, che è la funzione che serve per "combinare" i valori calcolati
- (b) **F**, che è la funzione da applicare a ciascun termine enumerato
- (c) **Next**, che è la funzione che ci dice come calcolare il prossimo valore da valutare dopo **a**
- (d) **ElementoNeutro**, che è l'elemento neutro rispetto l'operazione definita nella procedura **Operazione**
- (e) **a**, il primo elemento della sequenza da considerare
- (f) **b**, che è l'ultimo elemento da considerare

Accumula restituisce in output il valore accumulato combinando tramite l'**Operazione** specificata i valori ottenuti applicando la funzione **F** ad ogni elemento della sequenza che si ottiene applicando ad **a** la funzione **Next** sino a quando non si supera il valore **b**. Il **ValoreNeutro** viene restituito quando applicando **Next** ad **a** viene superato il valore **b**.

Si noti che a partire da questa funzione, si può definire la funzione **Sommatoria**, come segue:

```
1 def Sommatoria(F, a, Next, b):  
2     return Accumula(add, 0, F, a, Next, b) # NOTA: from operator import add
```

- (a) Come possiamo definire la produttoria?
 - (b) Scrivere una versione di **Accumula** che genera un processo ricorsivo.
 - (c) Scrivere una versione di **Accumula** che genera un processo iterativo.
2. Una funzione ancora più generale di **Accumula** è la funzione **FiltraAccumula** che introduce l'idea di avere un *filtro* sui valori da accumulare: vengono combinati dalla operazione specificata in input solo quegli elementi che soddisfano il predicato **Filter**. Implementare la seguente procedura:

```
1 def FiltraAccumula(Filter, Operazione, ElementoNeutro, F, a, Next, b):  
2     # DA COMPLETARE COME ESERCIZIO
```

Usare la funzione definita sopra per calcolare:

- (a) La somma dei quadrati dei numeri primi nell'intervallo $[a, b]$.
- (b) Il prodotto di tutti i numeri primi minori di n che sono primi rispetto ad n (ovvero tutti i numeri interi positivi $i < n$ tali che $MCD(i, n) = 1$).

3. Scrivere un costruttore **MakeRange(a, b)** che costruisce una lista (come catena di coppie, come visto a lezione) di numeri interi compresi tra **a** e **b**.

4. Scrivere una procedura **Append(As, Bs)** che prende in input due liste **As** e **Bs**, e restituisce un'unica lista che contiene prima tutti gli elementi della prima lista, e poi tutti quelli della seconda lista.

5. Scrivere una procedura **Scala(As, a)** che moltiplica ogni elemento della lista **As** per il termine **a**.

6. Scrivere una procedura **Quadrati(As)** che calcola il quadrato di ogni elemento della lista **As**.

7. (a) Scrivere una procedura **Map(F, As)** che applica la funzione **F** ad ogni elemento della lista **As**.
 (b) Riscrivere le due funzioni precedenti **Scala** e **Quadrati**, usando la funzione **Map**.

8. Scrivere una funzione **Filter(P, As)** che prende in input un predicato **P** e una lista **As** e restituisce in output la lista di elementi che soddisfano il predicato **P**. Usare la funzione **Filter** e la **IsPrime** dell'esercitazione 2, per ottenere la lista dei numeri primi compresi tra 1 e 30.

9. Scrivere una procedura **Reverse(As)** che prende in input una lista **As** e restituisce la stessa lista, ma con l'ordine degli elementi invertiti. Esempio: la lista **(1, (2, (3, None)))** diventa **(3, (2, (1, None)))**.