Esercizio 1

Definire in LISP una funzione ***nopred*** che ha per argomenti un predicato e una lista e che restituisce come valore la lista data come argomento SENZA gli atomi – a qualsiasi livello di profondità – che soddisfano il predicato. Attenzione ai NIL per mantenere la struttura della lista.

Esempio

prompt> **(nopred 'evenp ' (24 5 (7) d (((4))) 3 ())**
(5 (7) d ((()))) e ())

Esercizio 2

Dato il seguente programma Common LISP scriverne uno equivalente che utilizzi una lambda expression:

```
(defun f (x y)
    (let ((a (* (+ (* 2 x y)) 42))
          (b (+ 2 a))
          )
      (+ (* a (quadrato x))
          (+ b y)
          (* a b))))
```

Esercizio 3

Dato il seguente programma Common LISP:

```
(defun what (e n list)
    (cond ((zerop n) (cons e list))
            ((null list) (list e))
            (t (cons (car list) (what e (- n 1) (rest list))))))
```

descrivere il comportamento e calcolare il valore prodotto dalla valutazione della seguente applicazione:

prompt> **(what 'h 3 '(a b c d e))**

Esercizio 4

Definire in Common LISP la funzione ***variance*** che, presa una lista in ingresso, e un parametro ***&key*** chiamato, per l'appunto ***key***, restituisce la ***varianza*** dei valori presenti in ogni elemento (estratti con la funzione associata a ***key***). Potete assumere di avere a disposizione una funzione ***average*** che prende gli stessi argomenti di ***variance***, e che ritorna la media degli elementi nella lista in ingresso sempre con le stesse convenzioni sui parametri. La formula per la varianza di una variabile casuale ***X*** è $\mathbf{Var(X) = E[(X - \mu)^2]}$.

Esercizio 5

Dato il seguente programma LISP scriverne uno equivalente che utilizzi una ***let***:

```
(defun f (x y)
    (lambda (b a)
        (+ (* a (quadrato x))
            (* y b)
            (* a b)))
    (* 2 y)
    (* x (+ 1 y))))
```

Esercizio 6

Dare una definizione di ***cons-cell***. Dare una rappresentazione grafica in termini di cons-cell della seguente espressione simbolica:

1. **(a (b c) (d e) f)**
2. **(a (b) (c . 42) (d e))**
3. **(a ((b) c) ((d) . 42) e)**
4. **(a b (c 42 . f) (d ((e))))**
5. **(a (c) (d e))**

Esercizio 7

Definire in LISP una funzione che ha per argomento una lista e restituisce la lista con lo stesso numero di elementi, ottenuta applicando la funzione ***per-due*** ai soli atomi numerici dispari a qualsiasi livello di profondità. La funzione ***per-due*** raddoppia l'argomento che riceve se questo è un numero.

Esercizio 8

Data la seguente funzione LISP:

```
(defun splice-in-at (e n list)
    (cond ((zerop n) (cons e list))
        ((null list) (list e))
        (t (cons (first list)
                (splice-in-at e (- n 1) (rest list))))))
```

Descrivere il comportamento e calcolare il valore per gli argomenti:

'foo 3 '(4 6 2 9 42 -123)

Esercizio 9

Data la seguente funzione LISP:

```
(defun multiplex (functions value)
    (if (null functions) NIL
      (cons (funcall (first functions) value)
            (multiplex (rest functions) value))))
```

Descrivere il comportamento e calcolare il valore per gli argomenti:

1. **‘(consp integerp 1+ symbolp)**
2. **3.14**
3. **42**
4. **(list ‘list ‘zerop (lambda (x) (+ x 42)) ‘/)**

Esercizio 10

Data la seguente funzione in Common LISP:

```
(defun mistery (f list)
    (cond ((null list) NIL)
          ((null (rest list)) (list (funcall f (first list)))
          (t (append (list (funcall f (first list)))
                     (list (second list))
                     (misery f (rest (rest list)))))))
```

Descrivere il suo comportamento, ossia dire quali argomenti richiede e cosa restituisce come risultato.

Esercizio 11

Dato il seguente programma Common LISP scriverne uno equivalente che utilizzi ***let***:

```
(defun f (x y)
   (lambda (b a)
      (+ (* x a)
         (* y (quadrato b))
         (* a b)))
   (* x y)
   (* 4 y)))
```

Esercizio 12

Dato il seguente programma Common LISP scriverne uno equivalente che utilizzi una ***lambda expression***:

```
(defun f (x y)
    (let ((a (* (sin (* 2 x y)) pi))
          (b (* 2 a))
          )
      (+ (* a (quadrato x))
          (* y b)
          (* a b))))
```

Esercizio 13

Data la seguente funzione Common LISP:

```
(defun mistery (e n list)
    (cond ((zerop n) (cons e list))
          ((null list) (list e))
          (t (cons (cas list) (mistery e (- n 1) (rest list))))))
```

Descrivere il suo comportamento, ossia dire quali argomenti richiede e cosa restituisce come risultato.

Esercizio 14

Definire in LISP una funzione che ha per argomento un albero (ossia una lista) e lo restituisce, sostituendo, a qualsiasi livello di profondità, gli atomi numerici pari con il risultato dell'applicazione della funzione ***aggiungi-42***. La funzione ***aggiungi-42*** aggiunge il numero fondamentale al valore numerico.

Esercizio 15

Dato il seguente programma LISP scriverne uno equivalente che utilizzi ***lambda expression***:

```
(defun f (x y)
    (let ((a (+ x (* x y)))
          (b (* 4 y)))
        (lambda (x y)
          (+ (* x a)
              (* y (quadrato b))
              (* a b)))))
```

Esercizio 16

Dato il seguente programma LISP scriverne uno equivalente che utilizzi ***lambda expression***:

```
(defun f (x y)
    (let ((a (+ x (* x y)))
          (b (* 4 y)))
        (+ (* x a)
            (* y (quadrato b))
            (* a b))))
```

Esercizio 17

Data la seguente funzione LISP

```
(defun what (l1 l2)
    (cond ((null l1) 0)
          ((null l2) 0)
          ((memberp (car l1) l2) (add 1 (what (cdr l1) l2)))
          (t (what (cdr l1) l2))))
```

Descrivere il comportamento e calcolare il valore prodotto dalla valutazione della funzione applicata ai seguenti argomenti:

(2 3 4) (3 7)

Esercizio 18

Definire in LISP la funzione ***aggiungi1*** che ha per argomento una lista e restituisce la lista con lo stesso numero di elementi ottenuta aggiungendo un'unità ai soli atomi numerici dispari a qualsiasi livello di profondità.

Esempio

(aggiungi1 '(24 5 (7) d (((4))) 3 ()) → (24 6 (8) d (((4))) 4 ())

Esercizio 19c