definizione: La lista lineare è:

- la lista vuota

- o una coppia di elementi in cui il primo è un dato, il secondo una lista.

La definizione proposta è ricorsiva: nella definizione di lista compare la parola lista.
Ogni lista si può immaginare come una coppia di elementi in cui il primo è un dato,il secondo è la lista degli elementi rimanenti.

Con questa definizione la lista lineare può essere realizzata utilizzando le seguenti primitive:

costruttori:

```
Lista listaVuota();                 /* crea la lista vuota */
Lista listaCoppia(Dato, Lista);     /* crea la coppia che ha come primo elemento il dato, il
                                       secondo una lista */
```

selettori:

```
Dato listaPrimo(Lista );            /* restituisce il primo elemento di una coppia          */
Lista listaSecondo(Lista );         /* restituisce il secondo elemento, cioè una lista      */
```

E' necessaria una funzione che permetta di conoscere se una lista è vuota:

```
int isListaVuota(Lista );           /* 1 se la lista è vuota, 0 altrimenti */
```

Esempi.

1. Realizzare una funzione che restituisce una lista di interi. Gli interi vengono letti da tastiera.

```
Lista leggi( void ) {
      int i;

      printf("$$ ");
      if(scanf("%d",&i) == 1)
         return listaCoppia(i, leggi());
      return listaVuota();
}
```

La funzione è ricorsiva. Si basa sulla seguente definizione:

      se non c'è numero da leggere la lista è vuota
      altrimenti la lista è la coppia formata dal dato letto e dai rimanenti da leggere.

2. Stampare una lista.

La definizione adottata è la seguente:
      se la lista non è vuota

stampa il primo elemento
stampa la lista rimanente

Che diventa:

```
void stampa( Lista l ) {
    if(!isListaVuota( l )){
        printf("%d ", listaPrimo(l) );
        stampa( listaSecondo(l) );
    }
}
```

3. Stampa una lista in ordine inverso.

La definizione adottata è la seguente:

se la lista non è vuota
stampa gli elementi rimanenti
stampa il primo elemento

Che diventa:

```
void stamparev(Lista l) {
    if(!isListaVuota(l)){
        stamparev(listaSecondo(l));
        printf("%d ", listaPrimo(l) );

    }
}
```

4. Sommare tutti gli elementi di una lista di interi.

La definizione adottata è la seguente:

se la lista è vuota la somma è 0
altrimenti si somma il primo elemento della lista con la somma della lista rimanente.

```
int somma(Lista l) {
    if(isListaVuota(l)) return 0;
    return listaPrimo( l ) + somma( listaSecondo(l) );
}
```

5. Sommare tutti gli elementi pari di una lista di interi.

La definizione adottata è la seguente:

se la lista è vuota la somma è 0
se il primo elemento è pari

lo si somma alla somma degli elementi pari della lista rimanente
altrimenti si sommano gli elementi pari della lista rimanente.

```
int sommaPari(Lista l){
    if(isListaVuota(l)) return 0;
    if(pari(listaPrimo(l)))
    return listaPrimo(l) + sommaPari(listaSecondo(l));
    return sommaPari(listaSecondo(l));
}
```

6. Contare quanti elementi di una lista sono divisibili per 3.

La definizione adottata è la seguente:

se la lista è vuota i numeri divisibili per 3 sono 0
se il primo elemento è divisibile per 3 il numero degli elementi è
1 + il numero degli elementi divisibile per 3 della lista rimanente
altrimenti è il numero degli elementi divisibile per 3 della lista rimanente

```
int conta3(Lista l){
    if(isListaVuota(l)) return 0;
    if(mod3(listaPrimo(l)))
      return 1 + conta3(listaSecondo(l));
    return conta3(listaSecondo(l));
}
```