**ADT list()**

**Specifica Sintattica e Semantica**

**Tipi:**

* List: è l’insieme delle sequenze di n item, con N>= 0;
* Stringa;
* Interi;
* Item: insieme di elementi.

NULLITEM: è un elemento che viene restituito quando la precondizione di getFirst viene violata

**Operatori:**

* newlist() 🡪 list

1. Creazione e restituzione della nuova list vuota.
2. Creazione della lista attraverso l'operatore newList.

* Insertlist(list, item) 🡪 list

1. Calcolo della dimensione della playlist con l’uso dell'operatore per le liste sizeList;
2. Inserimento di un nuovo item in coda con l’uso dell'operatore insertList;
3. Restituzione di TRUE se l'inserimento è stato effettuato correttamente, di FALSE altrimenti produce una nuova lista con l’inserimento del nuovo item.

Precondizione: item =! NULLITEM per non inserire item nulli all’interno della lista.

Post condizione: list = < e1, e2, ..., en > con n >=0 🡪 list1 = < e1, e2, ..., en, el > con n >=0

* removeNode(list, intero) 🡪 list

1. Verifica sulla validità di pos. (compresa tra 0 e la grandezza della lista.);
2. Rimozione dell’item in posizione pos attraverso l'operatore per le liste removeList;
3. In caso di successo restituzione di TRUE in caso di fallimento restituzione di FALSE.

Precondizione: controllo se la lista ha degli item all’interno

Post condizione: list = < e1, e2, ..., en, el> con n >=0 🡪 list1 = < e1, e2, ..., en> con n >=0, la seguente operazione restituirà una lista senza l’elemento che abbiamo eliminato.

* scambio(list, intero) 🡪 list

1. Controllo sulla validità di pos;
2. Ottenimento temporaneo dell’item in posizione pos attraverso

la funzione getItem;

1. Rimozione dell'elemento in posizione pos dalla list attraverso la funzione removeList;
2. Inserimento dell’item prelevato precedentemente in posizione pos+1 della list;
3. Restituzione di 1 in caso di successo o di 0 in caso di fallimento.

Quest’operatore sostanzialmente scambia due nodi.

Precondizione: lista non vuota e posizione valida list= <e0, e1, e pos, e pos+1, en> n>=2 1<= pos <= n-1

Post condizione: l1= <e1,e2, e pos, e pos +1, en>

* outputlist(playlist) 🡪 playlist

1. Controllo dell'esistenza della lista;
2. Stampa della lista attraverso l'operatore outputList del modulo listc.

* reverselist(list, intero) 🡪 list

1. Creazione di una nuova lista attraverso l’uso dell’operatore newlist;
2. Finchè non si raggiunge la fine della lista di input avviene il prelevamento dell’elemento in posizione i della lista ricevuta da input mediante l’uso dell’operatore per le liste getitem;
3. Inserimento del nuovo item in coda attraverso l’uso dell’operatore per le liste insertlist;
4. Restituzione della nuova lista.

Quest’operatore sostanzialmente crea una nuova lista di ordine opposto rispetto a quello di input.

Precondizione: lista non vuota e list= <e0, e1, …,en, el> con n>=0

Post condizione: otterremo list1= <el, en, …, e1, e0> ovvero la lista di ordine opposto rispetto a quello di input.

* sublist(list, stringa) 🡪 list

1. Creazione di una nuova lista (attraverso l'operatore newPlist);
2. Finché la lista ricevuta in input non è vuota (ciclo da i = 0 a i = sizeList(p)) avviene il prelevamento dell'elemento in posizione i dalla lista ricevuta in input;
3. Confronto con la stringa ricevuta in input dell'elemento estratto (operatore eq del modulo item);
4. In caso coincidono aggiunta dell'elemento estratto in coda alla nuova lista (attraverso l'operatore addTail definito precedentemente nel modulo);
5. Restituzione della nuova lista.

Quest’operatore sostanzialmente crea una sotto lista.

Precondizione: lista non vuota e stringa in input valida

Post condizione: list= <e0, e1, …, en-1> n>=0 list1= <e0, e1, …, en-1> dove i campi che saranno presenti in questa sotto lista coincideranno con la stringa presa in input.