Fondamenti di Informatica L-B: Esercitazione 01

Autore: Andrea Urbini urbo83@tiscali.it

Matricola: 167064

Descrizione del problema

Si vuole affrontare la costruzione di un componente software di testing chiamato *MyTest*. Questa nuova classe deve testare approfonditamente tutte le funzioni della classe *ACMEList* fornita della software house, creando un oggetto (una lista) e interagendo con esso utilizzando i metodi della classe.

Inoltre si può costruire una nuova classe di nome *ACMEList2* in grado di ampliare quella già fornita, aggiungendo una funzione di ricerca di un elemento all'interno della lista.

Analisi

Innanzi tutto occorre spiegare com'è costituita una lista di n elementi interi: sono state definite due parti, la testa di dimensione 1 e la coda, vale a dire un'altra lista di dimensione n-1.
La soluzione della prima parte del problema è basata sulla realizzazione del componente software *MyTest* che deve svolgere le seguenti operazioni:

- creare un oggetto della classe *ACMEList;*
- inserire l'elemento 13;
- inserire l'elemento 303;
- visualizzare in standard output la lunghezza attuale della lista;
- svuotare la lista;
- inserire tutti i numeri multipli di 3 compresi fra 10 e 100;
- visualizzare ancora in standard output la lunghezza attuale della lista dopo gli inserimenti precedenti;
- calcolare la somma degli elementi contenuti nella lista e visualizzare in standard output tale somma.

La seconda parte problema implica la costruzione di una nuova classe (*ACMEList2*) basata sull'architettura di *ACMEList* alla quale è stato aggiunto il metodo *isPresent* che restituendo un dato di tipo boolean indica se un elemento è presente nella lista presa in considerazione. Inoltre è necessario l'ideazione di un piccolo test per questa nuova funzione inserito nella classe *MyTest.*

Progetto

Prima di tutto occorre una breve sintesi dei campi e dei metodi della classe *ACMEList*:

Campi:
private int nElements indica la lunghezza della lista
private int[] elements è l'array dove sono contenuti gli elementi della lista
Metodi:
public ACMEList() è il costruttore che crea un lista di lunghezza 10 (default)
public void addElement(int element) aggiunge un elemento alla coda della lista
public int getElementAt(int index) restituisce l'elemento nella posizione indicata
public void empty() svuota la lista
public int getLength() ritorna la lunghezza della lista
public boolean isEmpty() ritorna true se la lista è vuota

private void expand(int nElements) aumenta la lunghezza della lista

La prima operazione della classe *MyTest* è la creazione di una nuova variabile di nome *lista* e di tipo *ACMEList* a cui viene assegnato un nuovo oggetto di tipo *ACMEList*, invocando il costruttore della classe *ACMEList*. La lista è inizialmente vuota ma poi, tramite i metodi *AddElement* gli vengono inseriti gli elementi 13 e 303. Per visualizzare sullo standard output un numero occorre convertirlo in Stringa e richiamare la funzione *println* dell'oggetto *out* della classe *System*. Il numero da visualizzare è la lunghezza della lista fornita dal metodo *getLength*. Successivamente la lista va svuotata tramite *empty* e riempita con i multipli di 3 compresi tra 10 e 100 generati con un semplice algoritmo iterativo. Per calcolare la somma degli elementi della lista si crea una variabile somma e tramite un ciclo *for* da 0 alla lunghezza della lista si somma ogni elemento della lista alla variabile somma.

Il nuovo metodo della classe ACMEList2 ha la seguente segnature:

public boolean isPresent (int element)

Questo metodo svolge la ricerca di un elemento all'interno della lista confrontando ogni suo elemento con quello da trovare. Nel caso venga trovato si interrompe il ciclo di ricerca e viene ritornato *true*. Se dopo aver confrontato tutti gli elementi non viene trovato allora viene ritornato *false*. Se la lista è vuota, viene immediatamente ritornato *false*, in quanto nessun elemento è presente nella lista.

Dopo aver progettato la nuova funzione *isPresent* occorre testarla creando un nuovo oggetto di tipo *ACMEList2*, inserirgli dei numeri, e richiamare poi il metodo in questione chiedendogli di ricercare un elemento non presente nella lista e quello situato, ad esempio, nell'ultima posizione. Molto importante è richiamare la funzione *isPresent* quando la lista è vuota. Questa parte ho pensato di inserirla nella classe *MyTest*.

Implementazione

Di seguito riporto il codice dalla classe **ACMEList2.java** pressoché identica alla classe ACMEList2.java:

```
/**
 *  @author Andrea Urbini
 *
 */

public class ACMEList2 {

    // lunghezza corrente della lista
    private int nElements;

    // array ove sono contenuti gli elementi della lista
    private int[] elements;


    /**
     * Costruisce una lista, inizialmente vuota
     */
    public ACMEList2(){
        // creiamo un array iniziale atto a contenere 10 elementi
        elements = new int[10];
        // il numero di elementi a lista creata e' zero
        nElements = 0;
    }

    /**
     * Aggiunge un elemento in coda a questa lista
     *
     * @param element elemento da aggiungere
     */
    public void addElement(int element){

        // aggiungiamo l'elemento in coda alla lista
        // ovvero nella posizione mantenuta dalla variabile nElements
        elements[nElements]=element;
        // incrementiamo il numero corrente di elementi presenti
        nElements++;

        // nel caso in cui il numero di elementi
        // sia superiore alla lunghezza attuale dell'array
        // espandiamo l'array
```

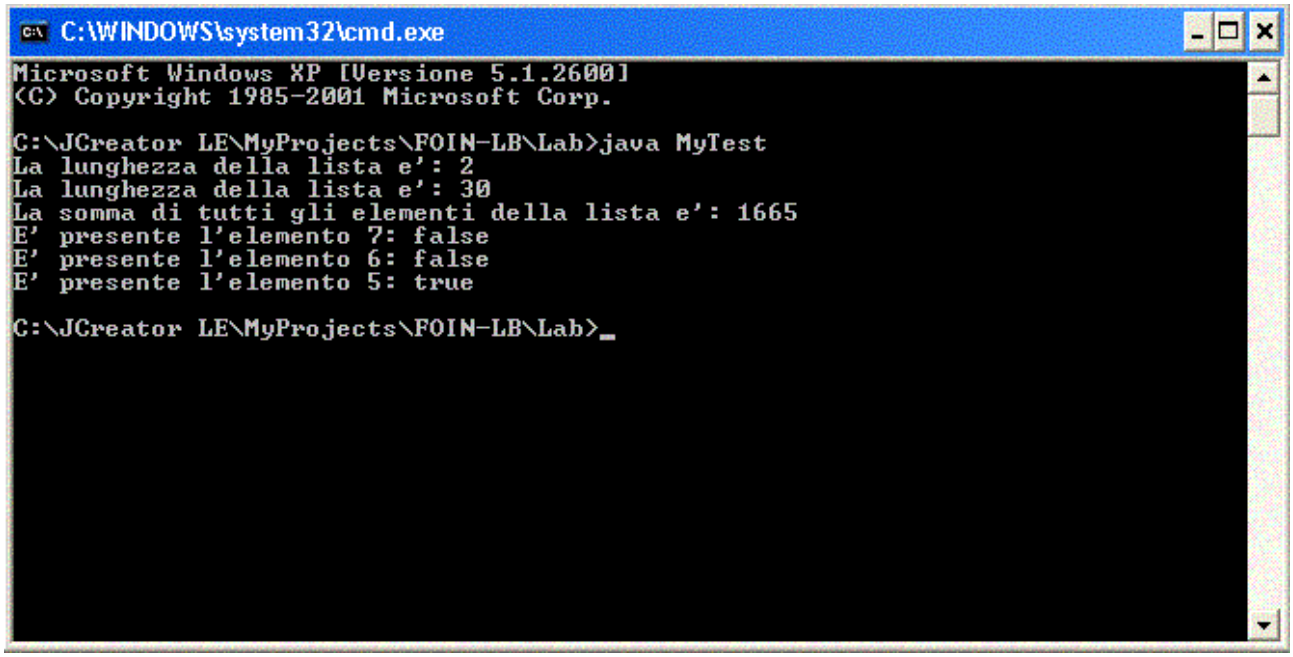
```
        if (nElements >= elements.length){
            expand(10);
        }
    }

    /**
     * Restituisce l'elemento di posizione specificata
     *
     * @param index  indice dell'elemento da restituire
     * @return  elemento della lista
     */
    public int getElementAt(int index){
        return elements[index];
    }

    /**
      * Elimina tutti gli elementi di questa lista, rendendola vuota.
      */
     public void empty(){
         nElements=0;
     }

    /**
     * Restituisce il numero degli elementi di questa lista (la sua lunghezza)
     *
     * @return numero elementi della lista
     */
    public int getLength(){
        return nElements;
    }

    /**
     * Testa se la lista e' vuota
     *
     * @return true se la lista e' vuota, ovvero contiene 0 elementi
     */
    public boolean isEmpty(){
        return nElements == 0;
    }


    /**
     * Espande l'array interno utilizzato per mantenere gli elementi della lista
     *
     * @param nElements numero di elementi da aggiungere
     */
    private void expand(int nElements){
        // creiamo un array di n elementi piu' lungo rispetto al precedente
        int[] newElements = new int[elements.length + nElements];
        // copiamo gli elementi dal vecchio al nuovo array
        for (int i=0; i < elements.length; i++){
            newElements[i]= elements[i];
        }
        // sostituiamo il vecchio array con quello nuovo
        elements = newElements;
    }

    /**
     * Ricerca l'elemento nella lista
     *
     * @param element elemento da cercare
     *
     * @return true se l'elemento e' presente
     */
    public boolean isPresent(int element){
       if (getLength()==0) {
               return false;
       }
       for(int i=0;i<=getLength();i++){
               if (element==getElementAt(i)){
                       return true;
               }
       }
       return false;
    }

}
```

Di seguito riporto il codice della classe **MyTest.java**:

```
/**
 * Componente software di testing della classe ACMEList e della classe ACMEList2
 * @author Andrea Urbini
 */

public class MyTest{

	public static void main(String[] args){
		//Creo un oggetto di tipo ACMEList
		ACMEList lista=new ACMEList();

		//Inserisco l'elemento 13 nella lista
		lista.addElement(13);

		//Inserisco l'elemento 303 nella lista
		lista.addElement(303);

		//Stampo la lunghezza della lista
		System.out.println("La lunghezza della lista e': "+lista.getLength());

		//Svuoto la lista
		lista.empty();

		//Inserisco nella lista i multipli di 3 compresi fra 10 e 100
		for (int i=4;i<=33;i++){
			lista.addElement(i*3);
		}

		//Stampo la lunghezza della lista
		System.out.println("La lunghezza della lista e': "+lista.getLength());

		//Calcolo la somma di tutti gli elementi e la stampo sullo standard output
		int somma=0;
		for (int i=0;i<lista.getLength();i++){
			somma+=lista.getElementAt(i);
		}
		System.out.println("La somma di tutti gli elementi della lista e': "+somma);

		//Creo un oggetto di tipo ACMEList2
		ACMEList2 lista2=new ACMEList2();

		//Controola se nella lista vuota è presente l'elemento 0 e stampa il risultato
		System.out.println("E' presente l'elemento 0: "+lista2.isPresent(0));

		//Riempo la lista di numeri da 1 a 5
		for (int i=0;i<=5;i++){
			lista2.addElement(i);
		}

		//Controlla se sono presenti alcuni elementi e stampa il risultato su standard output
		System.out.println("E' presente l'elemento 6: "+lista2.isPresent(6));
		System.out.println("E' presente l'elemento 5: "+lista2.isPresent(5));

	}
}
```

Casi d'uso

Questa è la schermata MS-DOS in cui si vedono i risultati stampati su schermo derivanti dall'esecuzione di MyTest.class.



Concetti e tecniche acquisiti

Questa esercitazione mi ha permesso conoscere l'esigenza di osservare con attenzione la documentazione fornita poiché durante il corso di Fondamenti L-A non avevo sentito questa necessità. Inoltre ho migliorato le mie conoscenze sugli oggetti e sulle classi che li generano grazie alla parte facoltativa dell'esercitazione che mi ha permesso di creare un metodo in un modo leggermente diverso rispetto a quello utilizzato nel corso precedente.