Fondamenti di Informatica T-1 (A.A. 2011/2012) - Ingegneria Informatica
Prof.ssa Mello
Prova Parziale d'Esame di Giovedì 12 Gennaio 2012 – durata 1h
Totale 12 punti, sufficienza con 7

Compito B

ESERCIZIO 1 (6 punti)

Data una lista di char `listChar` e una stringa ben formata `word`, si realizzi una funzione ITERATIVA

```
list select(list listChar, char* word){
```

che restituisca una nuova lista contenente i caratteri della stringa ben formata `word` che sono presenti nella lista `listChar`.

Ad esempio se `word = "costanza"` e `listChar = ['a', 'd', 'c']`, la funzione `select()` deve restituire la lista `['c', 'a', 'a']`, ovvero i soli valori della stringa ben formata `word` presenti in `listChar`.

La funzione `select()` dovrà essere implementata utilizzando le primitive dell'ADT lista; inoltre, si faccia uso della funzione `int member(char ch, list lista)` che restituisce 1 se il carattere `ch` è presente tra gli elementi della lista di char `lista`, 0 altrimenti. La funzione `member()` **NON** deve essere realizzata dal candidato. Ogni altra funzione (ad eccezione della funzione `member()` sopracitata) dovrà essere opportunamente specificata dal candidato. Si realizzi inoltre una semplice funzione `main()` di prova che invochi correttamente la funzione `select()` creata.

Nota: l'ordine degli elementi della lista restituita dalla funzione `select()` è ininfluente.

ESERCIZIO 2 (2 punti)

Si consideri la grammatica G con scopo S e simboli terminali {a, b, c, e, f, g}

```
S ::= W P | Q Z
Q ::= P W | W
P ::= Z Q | Z
Z ::= a | b | c
W ::= e | f | g
```

La stringa "agfb" appartiene al linguaggio generato da tale grammatica? In caso affermativo, se ne mostri la derivazione left-most.

ESERCIZIO 3 (3 punti)

Il seguente programma C compila correttamente? In caso affermativo, quali sono i valori stampati a tempo di esecuzione? (si motivi opportunamente la risposta data)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

char* translate(float* v, int* dim){
	char *res;
	int i = 0;
	float *temp;

	temp = v;
	while( *v >= 0 ){
		v++;
	}

	res = (char*) malloc(sizeof(char)*(v-temp));

	while( i < (v - temp) ){
		res[i] = 'a' + *(temp+i);
		i++;
	}
	*dim = i;

	return res;
}

int main() {
	float word[] = {1.2F, 0.1F, 1.8F, 0.0F, -5.3F, 6.2F};
	char* temp;
	int dim, i;
	temp = translate(word, &dim);
	for(i=0; i<dim; i++){
		printf("%c", temp[i]);
	}
	return 0;
}
```

ESERCIZIO 4 (1 punti)

Il candidato illustri brevemente le differenze tra allocazione statica e dinamica della memoria e mostri un breve esempio che metta a confronto queste due tecniche.

Soluzioni

ESERCIZIO 1

```
// solo per completezza (funzione non richiesta dal testo)
int member(char ch, list l){
      int found = 0;
      while( found == 0 && ! empty(l) ){
            if( ch == head(l) ){
                  found = 1;
            }
            else{
                  l = tail(l);
            }
      }
      return found;
}

// iterativa
list select (list l, char* word){
      list res;
      int ok;
      res = emptylist();
      while( *word  != '\0' ){
            ok = member(*word , l);
            if( ok == 1 ){
                  res = cons(*word , res);
            }
            word ++;
      }
      return res;
}

int main(){
      list l, res;
      char* parola = "costanza";
      l = cons('a', cons('d', cons('c', emptylist())));

      res = select(l,parola);
      while( ! empty(res) ){
            printf("%c ", head(res));
            res = tail(res);
      }

      return 0;
}
```

ESERCIZIO 2

La frase appartiene al linguaggio. In particolare, la si può ottenere tramite la seguente derivazione left-most:
S → QZ → PWZ → ZQWZ → aQWZ → aWWZ → agWZ → agfZ → agfb

ESERCIZIO 3

Il programma è corretto sintatticamente e la sua esecuzione produce la stampa:

baba

La funzione `main()` inizializza l'array di float `word` e poi invoca la funzione `translate()`.

La funzione `translate()` conta il numero di float dell'array `v` fino al primo elemento negativo escluso e poi alloca dinamicamente spazio di memoria sufficiente a contenere 4 char, ovvero un numero di char equivalente al numero di float contati fino a quel momento.

Il secondo ciclo `while` itera nuovamente lungo `v` (fino al primo elemento negativo escluso) e per ciascun float inserisce nell'area di memoria allocata dinamicamente un carattere minuscolo in relazione al valore del float (troncato implicitamente ad intero). Ad esempio, in caso di 1.8F il valore viene troncato all'intero 1 e nell'area di memoria viene inserito il carattere 'b' (ovvero 'a'+1). Infine la funzione `translate()` restituisce per valore un puntatore alla prima cella dell'area di memoria allocata dinamicamente e per riferimento la dimensione di tale area.

Il `main()` stampa sullo standard output i caratteri presenti nell'area di memoria allocata dinamicamente.