Programmazione – CdS in Matematica
Appello d'esame 4 Aprile 2008

Nome ……..
Cognome ……...
Matricola ……..

INDICARE SUBITO NOME, COGNOME, E MATRICOLA

NON è permesso (pena espulsione) usare la calcolatrice e consultare appunti e libri. Scrivere le risposte e commentare i programmi CHIARAMENTE (la chiarezza sarà un criterio determinante nella valutazione degli esercizi).

-------- PARTE 1 ----------

Esercizio 1.1

Scrivere l'output del seguente programma C. (Indicare con '_' underscore gli eventuali spazi e con '<n>' la riga completamente vuota)

```
#include<stdio.h>

int main() {
  char c = 'A';
  double x = 3.12, y = 0.01;
  printf("%d %d\n", c=='a', c<='D');
  printf("%d\n", x+y > x-y);

  char a = 'a', b='b'; /* N.B. ASCII('a') = 97 */
  int i=1, j=2;
  int val = (i==j) ? a-1 : b+1;
  printf("%d\n",val);

  char una_stringa[] = "questa e' una stringa lunga!";
  int n=11;
  for ( ; n>=0 ; n-=3)
    printf("%c", una_stringa[n]);
  printf("\n");
}
```

Esercizio 1.2

Implementare la seguente funzione:

1) `float ProdottoScalareArray(float a[], float b[], int n)`

Esercizio 1.3

Descrivere bene la sintassi e l'utilizzo dei costrutti MALLOC e CALLOC

Esercizio 1.4

Descrivere bene similitudini e differenze delle seguenti funzioni:

```
int Inc1(int a) { return a++; }
int Inc2(int *a) { return ++(*a); }
int Inc3(int *a) { return (*a)++; }
```

Esercizio 1.5

Descrivere le funzionalità (cosa calcola) e discutere le assunzioni che devono essere verificate sui parametri delle seguenti funzioni. Esemplificare dettagliatamente la loro invocazione.

A)

```
float A(float *a, int n) {
  int i;
  float m=0;
  for (i=0; i<n; i++)
    m += a[i];
  m /= n;
  return m;
}
```

B)

```
void B(char* pc) {
  if (!*pc) return;
  pc[0]=(*pc)++;
  B(pc+1);
}
```

C)

```
int C(int m[N][M]) {
 int q=0, i, j;
 for (i=0; i<N; i++)
  for (j=0; j<M; j++)
   q += m[i][j]*m[j][i];
 return q;
}
```

D)

```
void D(int k) {
  if (k==0) return;
  printf("%d",k);
  D(k-1);
  printf("%d",k);
}
```

-------- PARTE 2 ----------

Esercizio 2.1

Sia realizzi una funzione che approssimi il valore della serie esponenziale e^x (e che sia il piu' efficiente possibile!):

$$f(x,n) = \sum_{i=0}^{n} \frac{x^i}{i!}$$

```
float F(float x, int n)
```

Esercizio 2.2

Sia data la seguente struttura dati LISTA

```
struct ELEM {
  int dato;
  struct ELEM *prox;
};

typedef struct ELEM ElementoLista;
typedef ElementoLista *Lista;
```

Si implementino le seguenti funzioni:

```
Lista InserisciInTesta(Lista lista, int dato);
/* Inserisce un nuovo dato all'inizio della lista data come
parametro e ritorna la nuova lista*/

Lista CancellaTesta(Lista lista);
/* Cancella il primo elemento della lista e ritorna la nuova
lista*/

Lista CercaInLista(Lista lista, int dato);
/* Ritorna il puntatore al primo elemento dove occorre il dato o
NULL se non c'e' */

void StampaLista(Lista lista);
/* Stampa i dati contenuti nella lista */

int SommaLista(Lista lista);
/* Ritorna la somma degli elementi della lista */
```

Esercizio 2.3

Si dia un programma ricorsivo (ben commentato) che implementi l'algoritmo di ordinamento MERGESORT