1 Compito A

1.1 Esercizio 1 (7 punti)

Completare le parti mancati del seguente codice C con tutte le istruzioni necessarie per il corretto funzionamento del programma. Inserire dei commenti nel codice per descrivere le operazioni effettuate.

```c
/*
 * Soluzione di Eq. Non Lineari con il metodo
 * delle Secanti
 */

#include <stdio.h>
#include <math.h>

/* Funzione f - Eq. non lineare di cui si cercano gli zeri (da considerare
   implementata)*/
double f(double x);

int main()
{
  int N_max = 10;                  /* Numero massimo delle iterazioni */
  int N = 0;                       /* Numero di iterazioni */
  double x0 = 0.,x1 = 0.;          /* Estremi intervallo di integrazione */
  double err1 = 0., err2 = 0.;     /* Due modi di valutazione dell'errore */
  double err_max = 0.;             /* Errore massimo accettabile */
  double xk = 0.;                  /* Soluzione k-sima */

  /* Lettura parametri di input */

  printf("Approssimazione iniziale x0: ");
  scanf("%lf", &x0);
  printf("Approssimazione iniziale x1: ");
  scanf("%lf", &x1);
  printf("Errore massimo accettabile: ");
  scanf("%lf", &err_max);


/*  ----------------PARTE MANCANTE----------------------------
 *
 * inserire tutte le operazioni necessarie per completare il programma
 *
 */

  return 0;
}/* main */
```

1.2 Esercizio 2 (3 punti)

La seguente funzione calcola la norma N di un vettore e contiene degli errori all'interno del codice. Individuare e correggere gli errori.

```
1  double NORMA_N_VETTORE(double a[num], double ordine_norma, int num)
2  {
3    int i;
4    double norma = 0.;
5
6    for(i=0; i < num; i++)
7    {
8      norma = pow(fabs(a[i]), ordine_norma);
9    }
10
11   return pow(norma, ordine_norma);
12 }
```

2 Compito B

2.1 Esercizio 1 (7 punti)

Completare le parti mancati del seguente codice C con tutte le operazioni necessarie per il corretto funzionamento del programma. Inserire dei commenti nel codice per descrivere le operazioni effettuate.

```c
/*
 * Soluzione di Eq. Non Lineari con il metodo
 * di Newton (o metodo delle Tangenti)
 *
 */

#include <stdio.h>
#include <math.h>

/* Funzione f - Eq. non lineare di cui si cercano gli zeri (da considerare
    implementata)*/
double f(double x);
/* Funzione df - Derivata della funzione (da considerare implementata)*/
double df(double x);

int main()
{
  int N_max = 10;                 /* Numero massimo delle iterazioni */
  int N = 0;                      /* Numero di iterazioni */
  double err1 = 0., err2 = 0.;    /* Due modi di valutazione dell'errore */
  double err_max = 0.;            /* Errore massimo accettabile */
  double xk = 0.;                 /* Soluzione k-sima */
  double x0 = 0.;                 /* Approssimazione iniziale */

  /* Lettura parametri di input */
  printf("Errore massimo accettabile: ");
  scanf("%lf", &err_max);
  printf("Approssimazione iniziale: ");
  scanf("%lf", &x0);

/*  ----------------PARTE MANCANTE----------------------
 *
 * inserire tutte le operazioni necessarie per completare il programma
 *
 */

  return 0;
}/* main */
```

2.2 Esercizio 2 (3 punti)

La seguente funzione calcola la norma uno di una matrice e contiene degli errori all'interno del codice. Individuare e correggere gli errori.

```
1  double NORMA1_MATRICE(int a[num][num], int num)
2  {
3    int i,j;
4    int b;
5
6    for(i=0; i < num; i++) b[i]=1.0;
7
8    for(i=0; i < num; i++)
9    {
10     for(j=0; j < num; j++)
11     {
12       b[j] += abs(a[i][j]) ;
13     }
14   }
15
16   return MAX_ABS_VETTORE(num,b);
17 }
18
19 double MAX_ABS_VETTORE(int num, int a[num]);
```