Esercizio 1

(array)

- Creare un programma che legga da input un numero non noto a priori di interi (al più 10) terminati da 0. Tale sequenza può eventualmente contenere numeri ripetuti.
- Si memorizzi tale sequenza in un vettore di opportuna dimensione.
- Si stampino a video tutti i numeri per cui il successivo nel vettore è pari al numero stesso

Esercizio 1 - Soluzione

(array)

```
#include <stdio.h>
#include <stdlib.h>
#define DIM 10

int main(void) {
    int num, size, i;
    int values[DIM];

    size = 0;
    do {
          printf("Inserisci un numero:");
          scanf("%d", &num);
          if (num!=0 && size<DIM) {
                  values[size] = num;
                  size++;
          }
    } while (num!=0 && size<DIM);

    for (i=0; i<size-1; i++)
          if (values[i] == values[i+1])
                  printf("%d ", values[i]);
    system("PAUSE");
    return (0); }
```

Esercizio 2
(array)

- Creare un programma che legga da input un numero non noto a priori di interi positivi (al più 10) terminati da 0.
- Qualora l'utente inserisca dei valori negativi, tali valori devono essere scartati e non considerati
- Si memorizzi tale sequenza in un vettore di opportuna dimensione.
- Si stampino a video tutti i numeri che sono allocati nel vettore in posizioni il cui indice è uguale al numero stesso

Esercizio 2 - Soluzione

(array)

```
#include <stdio.h>
#include <stdlib.h>
#define DIM 10

int main(void) {
    int num, size, i;
    int values[DIM];

    size = 0;
    do {
          printf("Inserisci un numero:");
          scanf("%d", &num);
          if (num>0 && size<DIM) {
                  values[size] = num;
                  size++;
          }
    } while (num!=0 && size<DIM);

    for (i=0; i<size; i++)
          if (values[i] == i)
                  printf("%d ", values[i]);
    system("PAUSE");
    return (0); }
```

Esercizio 3

(array)

- Creare un programma che legga da input un numero non noto a priori di interi (al più 10) terminati da 0.
- Si memorizzi tale sequenza in un vettore di opportuna dimensione, rispettando l'ordine con cui i valori sono stati inseriti.
- Si memorizzino poi in un secondo vettore i valori del primo, avendo cura di inserirli in ordine inverso
- Si stampi infine il vettore coi valori in ordine invertito

Esercizio 3 - Soluzione

(array)

```
#include <stdio.h>
#include <stdlib.h>
#define DIM 10

int main(void) {
    int num, size, i, j;
    int values[DIM], inv[DIM];
    size = 0;
    do {
          printf("Inserisci un numero:");
          scanf("%d", &num);
          if (num>0 && size<DIM) {
                    values[size] = num;
                    size++;
          }
    } while (num!=0 && size<DIM);

    j=0;
    for (i=size-1; i>=0; i--) {
          inv[j] = values[i];
          j++;
    }
    for (i=0; i<size; i++) printf("%d ", inv[i]);
    system("PAUSE");
    return (0);
}
```

Esercizio 4
(array)

- Realizzare un programma che legga da input una sequenza di interi positivi, terminati da 0.
- Tali numeri devono essere memorizzati in un array (di dimensione massima 10)
- Il programma quindi provveda a stampare a video tutti i numeri pari che sono memorizzati nell'array in una posizione con indice pari

- Estensione: si abbia cura di verificare che siano immessi numeri fino al limite di 10 elementi; dopo tale limite, il programma stampi un messaggio di errore, finchè la sequenza non è terminata da 0.
- Estensione: riprogettare il programma spostando la fase di acquisizione dei dati e l'algoritmo di stampa in due funzioni apposite.

Esercizio 4 - Soluzione
(array)

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

#define LIMIT 10

int main(void) {
    int i, size, num;
    int numeri[LIMIT];
    size = 0;
    do {
          printf("Inserire un numero: ");
          scanf("%d", &num);
          if (num>0 && size<LIMIT ) {
                    numeri[size] = num;
                    size++;
          }
          else if (size>=LIMIT){
                    printf("Spazio esaurito...\n");
          }
    } while (num != 0);
    for (i=0;i<size; i++)
          if ((i%2)==0 && (numeri[i]%2)==0)
                    printf("Numero all'indice %d: %d\n", i, numeri[i]);
    system("PAUSE");
    return (0); }
```

Esercizio 5
(array)

Si scriva un programma che
1) richieda all'utente un valore **V** di soglia;
2) successivamente prenda in ingresso una sequenza di reali positivi terminata da 0 (massimo 10), e memorizzi in un vettore di float **M** (di dimensione fisica 10) SOLO i valori maggiori di V;
3) infine crei un secondo vettore **MED** in cui l'elemento i-esimo è calcolato come la media tra l'elemento i-esimo del vettore M e il valore V.

Esercizio 5
(array)

Esempio: l'utente inserisce il valore 2.5 di soglia.
Poi inserisce la sequenza

```
1.3   4     5.2     9.5    2.2     1     0
```

Nel vettore M vengono quindi memorizzati solo

	0	1	2
M	4	5.2	9.5

Infine, il programma deve creare un secondo vettore MED in cui l'elemento i-esimo e' calcolato come la media tra l'elemento i-esimo del vettore M e il valore V.

	0	1	2
MED	3.25	3.85	6.0

Esercizio 5 - Soluzione
(array)

```
#include <stdio.h>

int main() {
        float V, num;
        float M[10], MED[10];
        int i=0, j=0;

         printf("Inserisci la soglia");
         scanf("%f",&V);
         do {
                printf("Inserisci elemento");
                scanf("%f",&num);
                if (num > V) {
                        M[i]=num;
                        i++;
                }
        } while ( (num!=0) && (i<10) ); // i=dimensione logica

        for(j=0; j<i; j++) {
                MED[j] = (M[j] + V)/2;
                printf("%f\n", MED[j]);
        }
        return 0;
}
```

Esercizio 6
(array)

Si scriva un programma che prende in ingresso una sequenza di massimo 10 reali positivi terminata da 0, e la memorizzi in un vettore di float **`NUM`**.

Il programma deve creare un secondo vettore **`MEDIE`** in cui l'elemento i-esimo è calcolato come la media tra l'elemento i-esimo del vettore **`NUM`** e il suo successivo. *Ovviamente la dimensione logica di medie sarà pari alla dimensione logica di **`NUM`** meno 1.*

Esercizio 6
(array)

Esempio: l'utente inserisce la sequenza

```
1.3   4     5.2     9.5    2.2     1     0
```

NUM	1.3	4	5.2	9.5	2.2	1	0

MEDIE	2.65	4.60	7.35	5.85	1.6	0.5

Esercizio 6 - Soluzione
(array)

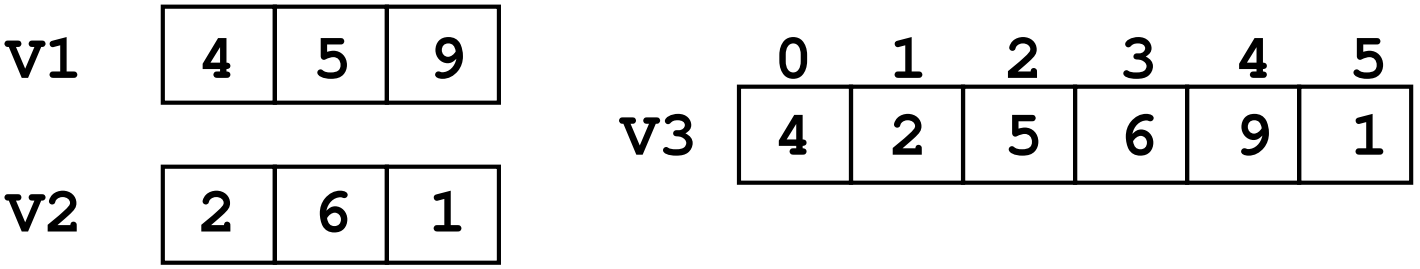
```
#include <stdio.h>

int main() {
        int i=0, j=0;
        float NUM[10], MED[10];

        do {
                printf("Inserisci elemento");
                scanf("%f",&NUM[i]);
                i++;
        } while ( (NUM[i-1]!=0) && (i<10) );

        for(j=0; j<i-1; j++) {
                MED[j] = (NUM[j+1] + NUM[j])/2;
                printf("%f\n", MED[j]);
        }
        return 0;
}
```

Esercizio 7
(array)

Scrivere un programma che

1) Legga da input due vettori **V1** e **V2** di interi di dimensione N;

2) costruisca un terzo vettore **V3** di dimensione 2N i cui elementi di posizione pari siano gli elementi del primo vettore e gli elementi di posizione dispari siano gli elementi del secondo vettore.



Esercizio 7 - Soluzione
(array)

```
#include <stdio.h>
#define N 3

int main() {
        int i, v1[N],v2[N],v3[2*N];

        for (i=0; i<N; i++) {
                printf("Inserisci elemento %d del primo vettore", i);
                scanf("%d",&v1[i]);
        }
        for (i=0; i<N; i++) {
                printf("Inserisci elemento %d del secondo vettore", i);
                scanf("%d",&v2[i]);
        }
        for (i=0; i<N; i++) {
                v3[2*i] = v1[i];
                v3[2*i+1] = v2[i];
        }
        for (i=0; i<2*N; i++)
                printf("%d  ", v3[i]);
        return 0; }
```

Esercizio 8

(array)

Scrivere un programma che, dato un vettore **`NUM`** di **`N`** interi positivi inseriti dall'utente, ne produca due **`PAR`** e **`DIS`** contenenti, rispettivamente, i numeri pari e dispari del vettore iniziale.

Si controlli che i numeri inseriti dall'utente siano positivi.

Per verificare se un numero è pari, si scriva una funzione che restituisca 1 se il numero è pari, 0 altrimenti

```
int pari(int n);
```

Esercizio 8

(array)

Esempio: l'utente inserisce la sequenza

4 2 −4 5 6 −9 1 6

↳ *scartati* ↲

NUM	4	2	5	6	1	6

PAR	4	2	6	6		

DIMENSIONE LOGICA 4

DIS	5	1				

DIMENSIONE LOGICA 2

Esercizio 8 - Soluzione

(array)

```
#include <stdio.h>
#define N 8

int pari(int n){
        if (n%2 == 0) return 1;
        else return 0;
}

void main()
{int num[N], par[N], dis[N];
 int i,ivp=0,ivd=0;  /* ivp = indice vett pari e ivd = indice vett
dispari*/

 for (i=0; i<N; i++)  /*lettura vettore num */
         do {printf("inserire intero positivo del vettore NUM ");
             scanf("%d", &num[i]);
            }
          while (num[i] < 0);
```

Continua...

Esercizio 8 - Soluzione

(array)

```
for (i=0; i<N; i++) /* trasferimento pari in PAR e dispari in DIS*/
        if (pari(num[i])){
            par[ivp] = num[i];
            ivp++;}
        else { dis[ivd] = num[i];
               ivd++;
              }

/* stampa dei vettori PAR e DIS
   ATTENZIONE !!!!! La dimensione logica di PAR e' ivp mentre
                    la dimensione logica di DIS e' ivd  */

 for (i=0; i<ivp; i++)
       printf("%d  ", par[i]);

 printf("\n");

 for (i=0; i<ivd; i++)
       printf("%d  ", dis[i]);

}
```

Esercizio 9
(array)

- Creare un programma che legga da input una sequenza di interi positivi. In particolare, l'utente inserisce un numero iniziale con cui specifica quanti numeri è intenzionato ad inserire (al max 10). Di seguito poi inserisce i numeri, tutti in ordine strettamente crescente.

- In fase di lettura, il programma controlli che ogni numero sia effettivamente maggiore del precedente (si scartino i valori che non rispettano tale criterio).

- In un secondo vettore si calcoli la differenza percentuale tra un valore ed il successivo (data dalla differenza tra i due valori, divisa poi per il primo valore e moltiplicato il tutto per 100)
- Si richieda poi all'utente un valore di soglia (in percentuale) , e si stampino a video tutte le coppie di valori il cui aumento dal primo al secondo valore risulta essere, in percentuale, maggiore della soglia specificata

Esercizio 9 - Soluzione

(array)

```
#include <stdio.h>
#include <stdlib.h>
#define DIM 10

int main(void) {
    int num, size, i, prec;
    int values[DIM];
    float soglia;
    float ratio[DIM];

    do {
          printf("Quanti numeri vuoi inserire ? (MAX 10) ");
          scanf("%d", &size);
    } while (size < 0 || size > DIM);

    prec = -1;
    for (i=0; i<size; ) {
          printf("Inserisci un numero:");
          scanf("%d", &num);
          if ( num > prec ) {
                    values[i] = num;
                    prec = num;
                    i++;
          }
    }
    …
```

Esercizio 9 - Soluzione

(array)

```
    …

    for (i=0; i<size-1; i++)
          ratio[i] = ( (values[i+1]-values[i]) / ((float) values[i])) * 100;

    printf("Inserire soglia: ");
    scanf("%f", &soglia);
    for (i=0; i<size-1; i++)
          if (ratio[i] > soglia)
                printf("%d %d\n", values[i], values[i+1]);

    system("PAUSE");
    return (0);
}
```

Esercizio 10
(array)

- Si vogliono elaborare alcuni dati metereologici, riguardanti alcune località sciistiche. Per ogni località, un'utente inserisce il codice della località e i cm. di manto nevoso (entrambi interi). Il codice 0, inserito come località, indica che l'utente non vuole inserire altri dati.

- Il programma deve memorizzare tali dati in due appositi vettori (uno per le località ed uno per la neve caduta)

- Il programma deve poi stampare a video i codici di tutte le località che risultino avere un manto nevoso inferiore alla media, calcolata sui valori inseriti

Esercizio 10 - Soluzione

(array)

```
#include <stdio.h>
#include <stdlib.h>
#define DIM 10

int main(void) {
    int cod_loc, size, i, tot;
    int loc[DIM], neve[DIM];
    float media;

    size = 0;
    do {
         printf("Inserisci codice localita': ");
         scanf("%d", &cod_loc);
         if ( cod_loc != 0 && size < DIM ) {
                  loc[size] = cod_loc;
                  printf("Manto nevoso (cm.): ");
                  scanf("%d", &(neve[size]));
                  size++;
         }
    } while (cod_loc!=0 && size<DIM);

    …
```

Esercizio 10 - Soluzione
(array)

```
    …

    tot = 0;
    for (i=0; i<size; i++)
          tot = tot + neve[i];
    media = tot / ((float) size);

    for (i=0; i< size; i++)
          if (neve[i]<media)
                printf("localita' con poca neve: %d\n", loc[i]);

    system("PAUSE");
    return (0);
}
```