COGNOME E NOME	MATRICOLA	A
AULA	POSTAZIONE	

Istruzioni per lo svolgimento della prova

La prova consta di quattro quesiti, i primi due da svolgere al calcolatore il terzo e il quarto su carta.

La soluzione del quesito 1 in modo sufficiente (cioè il programma deve compilare ed eseguire correttamente) è condizione necessaria per il superamento della prova.

I file sorgenti (.c) relativi alla soluzione dei quesiti 1 e 2 devono essere consegnati su Studium nella sezione elaborati in un unico file compresso (preferibilmente .zip o .rar) contenente entrambi i due file. Inserire all'inizio dei file sorgenti il proprio nome e numero di matricola come commento.

Quesito 1

Questo esercizio prevede l'uso di codice esistente da integrare e completare in modo da risolvere il problema proposto. Lo studente **deve utilizzare il codice proposto senza alcuna modifica** intervenendo solo nelle parti richieste ed integrandolo se necessario.

Perché l'esercizio sia sufficiente è necessario che il codice compili **senza errori**, il programma esegua senza **errori run-time**, e fornisca il risultato previsto.

```
#include <stdio.h>
#include <stdlib.h>
/*
La funzione legge un vettore di numeri interi fino all'inserimento del valore 0 o il raggiungimento del numero massimo
di elementi n. La funzione restituisce il numero dei valori effettivamente letti
*/
int lettura_dati(int* v, int n) {
        for (int i = 0;i < n; ++i) {
                scanf("%d", &v[i]);
                if (v[i] == 0)
                        return i;
        }
        return n;
}
/*
La funzione dato un vettore di ingresso che contiene valori diversi da zero restituisce un vettore contenente tutti i
valori positivi presenti nel vettore di ingresso.
Il vettore deve essere allocato all'interno della funzione della dimensione esattamente uguale al numero di valori
positivi presenti nel vettore di ingresso
la funzione restituisce (return) il vettore generato
        dim è la dimensione del vettore di ingresso
        vettoreIngresso è il vettore di ingresso
        *dim2 consente di restituire la dimensione del vettore
*/
int *estrae_numeri_positivi(int* vettoreIngresso, int dim, int *dim2) {
        // Il codice deve essere inserito dallo studente
}
/*
La funzione stampa su console il contenuto di un vettore
*/
void stampa( /* i parametri devono essere inseriti dallo studente */) {
        // Il codice deve essere inserito dallo studente
}
int main(void) {
        int *v1;                // vettore allocato dinamicamente
        int *v2;                // vettore allocato dinamicamente
        int dim;                // dimensione del vettore v1
        int dim2;


// Inserire il codice per implementare la lettura da tastiera del numero massimo di elementi inseribili nel vettore v1

// Lettura da tastiera del primo vettore utilizzando la funzione lettura_dati
        dim = lettura_dati( /* lo studente inserisca i parametri corretti */);

// Lo studente deve chiamare la funzione estrae_numeri_positivi salvando i valori nel vettore v2 allocato
// all'interno della funzione.

// lo studente, usando la funzione stampa, stampa il risultato contenuto in v2
        stampa(/* inserire i parametri*/);
// disallocazione
}
```

I QUESITI SUCCESSIVI NON VERRANNO CORRETTI SE IL PROGRAMMA DEL QUESITO 1 NON FUNZIONA CORRETTAMENTE COME DESCRITTO ALL'INIZIO DELL'ESERCIZIO.

Quesito 2

Si vuole realizzare una applicazione che consente di gestire una prova di esame. Lo studente può effettuare due tipi di prenotazione

- prova pratica
- registrazione

Le prenotazioni vengono raccolte in un vettore di liste, la lista di posto 0 conterrà le prenotazioni per la prova pratica, la lista di posto 1 le prenotazioni per la la registrazione.

Per ogni prenotazione saranno memorizzate le seguenti informazioni: numero di matricola, nome, cognome

Lo studente implementi un programma ANSI C che consente di effettuare le prenotazioni e memorizzare nella struttura dati precedentemente descritta gli esami superati.

Scrivere un opportuno ***main*** in cui sia previsto un menu di scelta delle operazioni richieste. Tutti i valori necessari al funzionamento devono essere passati utilizzando parametri, **non è permesso l'uso di variabili globali.**

1. Prenotazione di un nuovo studente da console. I dati da inserire sono il tipo di prova, il numero di matricola, il cognome, il nome e il voto nel caso di registrazione. Nel caso della prenotazione per la prova pratica il valore del voto è -1;
2. Funzione che restituisce un vettore che contiene il numero di prenotati per ciascuna prova;
3. RisultatoProvaPratica, la funzione acquisisce il nome di uno studente e un voto, se lo studente è presente nella lista della prova pratica lo trasferisce dalla lista della prova pratica (cancellandolo) nella lista registrazione inserendo il voto inserito;
4. Media, la funzione calcola il voto medio degli esami da registrare

Quesito 3

Dato il valore -45 calcolare il valore codificato con 16 bit utilizzando la codifica in modulo e segno e in complemento a due

Modulo e Segno	Complemento a due

Quesito 4

Descrivere una funzione che dato un albero binario di ricerca di elementi interi, restituisca una lista ordinata contenente i valori positivi presenti nell'albero. Si supponga di avere a disposizione il tipo lista e la funzione di inserimento in lista ordinata.

COGNOME E NOME	MATRICOLA	B
AULA	POSTAZIONE	

Istruzioni per lo svolgimento della prova

La prova consta di quattro quesiti, i primi due da svolgere al calcolatore il terzo e il quarto su carta.

La soluzione del quesito 1 in modo sufficiente (cioè il programma deve compilare ed eseguire correttamente) è condizione necessaria per il superamento della prova.

I file sorgenti (.c) relativi alla soluzione dei quesiti 1 e 2 devono essere consegnati su Studium nella sezione elaborati in un unico file compresso (preferibilmente .zip o .rar) contenente entrambi i due file. Inserire all'inizio dei file sorgenti il proprio nome e numero di matricola come commento.

Quesito 1

Questo esercizio prevede l'uso di codice esistente da integrare e completare in modo da risolvere il problema proposto. Lo studente **deve utilizzare il codice proposto senza alcuna modifica** intervenendo solo nelle parti richieste ed integrandolo se necessario.

Perché l'esercizio sia sufficiente è necessario che il codice compili **senza errori**, il programma esegua senza **errori run-time**, e fornisca il risultato previsto.

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

void LeggiBuffer(int* buf, int n)
{
  for (int i=0; i<n; i++) {
    printf("Inserisci elemento di posizione %d: ", i);
    scanf("%d", &buf[i]);
  }
}

/* Restituisce un array di interi ripetendo l'elemento bc[i+1] un numero di volte pari a bc[i]. Per esempio, se
bc = {5,2,4,3}, la funzione restituirà l'array {2,2,2,2,2, 3,3,3,3} */
int* RunLengthDecoding(int* bc, int nbc, int* nbd) {...}

// Visualizza gli elementi di un array passato come parametro
void Visualizza(...) {...}

int main(void)
{
  int *buffer_c, *buffer_d;
  int dim_c, dim_d;

  printf("Inserisci la dimensione del buffer compresso (la dimensione deve essere un numero pari): ");
  scanf("%d", &dim_c);
  // Assicurarsi che dim_c sia un numero pari e se non lo è ripetere l'inserimento

  buffer_c = ...; // Alloca un array di 'dim_c' elementi interi
  LeggiBuffer(...); // Acquisisce gli elementi in buffer_c

  buffer_d = RunLengthDecoding(...);

  Visualizza(...); // Visualizza buffer_d

  ...; // Disalloca buffer_c
  ...; // Disalloca buffer_d

  return 0;
}
```

I QUESITI SUCCESSIVI NON VERRANNO CORRETTI SE IL PROGRAMMA DEL QUESITO 1 NON FUNZIONA CORRETTAMENTE COME DESCRITTO ALL'INIZIO DELL'ESERCIZIO.

Quesito 2

Un dispositivo per l'analisi del traffico consente di identificare il tipo di veicolo (autoveicolo, motoveicolo, autocarro) che in un generico istante sta attraversando una certa corsia in una strada. Per ogni attraversamento, il dispositivo fornisce le seguenti informazioni: tipo veicolo, orario, corsia. Ogni volta che il dispositivo rileva il passaggio di un veicolo in una certa corsia, memorizza, in un array di liste, l'orario ed il tipo di autoveicolo. L'array ha tanti elementi quante sono le corsie.

Lo studente implementi un programma ANSI C che consente mantenere lo storico del traffico su una strada a 3 corsie. Scrivere un opportuno ***main*** in cui sia previsto un menu di scelta delle operazioni richieste. Tutti i valori necessari al funzionamento devono essere passati utilizzando parametri, **non è permesso l'uso di variabili globali.**

1. Acquisizione da file delle informazioni del traffico (guardare esempio in calce al quesito) e memorizzazione nell'array di liste.
2. Funzione che restituisce un vettore che contiene il numero di veicoli che hanno percorso ogni corsia.
3. Funzione che data una fascia oraria (passata come parametro) restituisca (e non visualizzi) il numero di veicoli per ogni tipo che hanno percorso la strada in quella fascia oraria.
4. Funzione che restituisca il numero complessivo di veicoli che hanno percorso la strada in ognuna delle 24 fasce orarie definite come di seguito: 00-01, 01-02, 02-03, …, 22-23, 23-24

Esempio di file

```
Autoveicolo    08:00 1
Autoveicolo    08:01 2
Motoveicolo    08:01 1
Autocarro      08:03 2
Autoveicolo    08:03 3
```

Quesito 3

Descrivere il ruolo dei registri in una CPU.

Quesito 4

Costruire l'albero binario di ricerca assumendo il seguente ordine di inserimento: 8, 9, 1, 3, 2, 4, 15, 12, 14, 21. Visualizzare la sequenza dei nodi visitati se si effettua la visita anticipata (radice, sinistra, destra).

COGNOME E NOME	MATRICOLA	C
AULA	POSTAZIONE	

Istruzioni per lo svolgimento della prova

La prova consta di quattro quesiti, i primi due da svolgere al calcolatore il terzo e il quarto su carta.

La soluzione del quesito 1 in modo sufficiente (cioè il programma deve compilare ed eseguire correttamente) è condizione necessaria per il superamento della prova.

I file sorgenti (.c) relativi alla soluzione dei quesiti 1 e 2 devono essere consegnati su Studium nella sezione elaborati in un unico file compresso (preferibilmente .zip o .rar) contenente entrambi i due file. Inserire all'inizio dei file sorgenti il proprio nome e numero di matricola come commento.

Quesito 1

Questo esercizio prevede l'uso di codice esistente da integrare e completare in modo da risolvere il problema proposto. Lo studente **deve utilizzare il codice proposto senza alcuna modifica** intervenendo solo nelle parti richieste ed integrandolo se necessario.

Perché l'esercizio sia sufficiente è necessario che il codice compili **senza errori**, il programma esegua senza **errori run-time**, e fornisca il risultato previsto.

```
#include <stdio.h>
#include <stdlib.h>
void InserisciVettore(int v[], int dim) {
        int i;
        for (i = 0; i < dim; i++) {
               printf("Inserisci l'elemento di indice %d: ", i);
               scanf("%d", &v[i]);
        }
}
/*
La funzione PostiPari, dato il vettore v, deve inserire gli elementi di v di posizione pari in un nuov vettore che deve essere allocato internamente alla funzione e restituito al main. La dimensione di questo nuovo vettore è dimp e va determinata internamente alla funzione.
*/
int* PostiPariDispari(int v[], int dim, int *dimp){
        . . .
}
/*
Visualizza i dati del vettore fornito in ingresso
*/
...StampaVettore(...);

int main(void) {
        int *v1;          /* primo vettore – allocato dinamicamente */
        int *v2;          /* secondo e terzo vettore */
        int dim1;         /* dimensione del primo vettore */
        int dim2;         /* dimensione del secondo vettore */
        printf("Inserisci il numero di elementi del vettore: ");
        scanf("%d", &dim);
        /* alloca il primo vettore */
        ...
        /* inserisce gli elementi del vettore */
        InserisciVettore(...);
        /* chiama la funzione PostiPari */
        ...= PostiPari(...);
        /* visualizza il vettore iniziale ed il vettore ottenuto come risultato */
        StampaVettore(...);
        StampaVettore(...);
        /* disalloca i vettori */
```

```
        ...
        return 0;
}
```

I QUESITI SUCCESSIVI NON VERRANNO CORRETTI SE IL PROGRAMMA DEL QUESITO 1 NON FUNZIONA CORRETTAMENTE COME DESCRITTO ALL'INIZIO DELL'ESERCIZIO.

Quesito 2

Si vuole sviluppare una applicazione per la gestione settimanale dei propri programmi tv preferiti. Le informazioni dei programmi sono mantenute in un vettore di 7 liste (una per ciascun giorno della settimana) e includono:

– nome canale televisivo

– titolo programma

– orario inizio (stringa nel formato hh:mm)

– durata (specificata in minuti)

Lo studente implementi un programma ANSI C dotato di un opportuno ***main*** in cui sia previsto un menu di scelta delle operazioni richieste e quant'altro necessario a fare funzionare il programma comprese le eventuali operazioni di ingresso/uscita. Tutti i valori necessari al funzionamento devono essere passati utilizzando parametri, **non è permesso l'uso di variabili globali.**

Implementare le seguenti funzioni:

1. Acquisizione da console delle informazioni di un programma televisivo e memorizzazione nel vettore di liste.
2. Funzione che, dato in ingresso un giorno della settimana, calcoli e restituisca la durata massima dei programmi di quel giorno ed il canale in cui quel programma viene trasmesso.
3. Funzione che, per ogni giorno della settimana, calcoli e restituisca in un opportuno vettore il numero di programmi che hanno una durata inferiore ad una certa soglia fornita come parametro di ingresso.
4. Funzione che, dato in ingresso un canale televisivo, inserisca in una lista tutti i programmi di quel canale televisivo che hanno inizio dopo un certo orario fornito come parametro di ingresso.

Quesito 3

Descrivere il ruolo del bus dati, bus indirizzi, e bus di controllo in un calcolatore elettronico.

Quesito 4

Si costruisca un albero binario di ricerca assumendo il seguente ordine di inserimento dei dati: 52, 76, 33, 45, 48, 7, 10, 92, 84, 38. Visualizzare la sequenza dei nodi visitati nel caso in cui si effettua una visita in post-ordine (visita differita).

COGNOME E NOME	MATRICOLA	D
AULA	POSTAZIONE	

Istruzioni per lo svolgimento della prova

La prova consta di quattro quesiti, i primi due da svolgere al calcolatore il terzo e il quarto su carta.

La soluzione del quesito 1 in modo sufficiente (cioè il programma deve compilare ed eseguire correttamente) è condizione necessaria per il superamento della prova.

I file sorgenti (.c) relativi alla soluzione dei quesiti 1 e 2 devono essere consegnati su Studium nella sezione elaborati in un unico file compresso (preferibilmente .zip o .rar) contenente entrambi i due file. Inserire all'inizio dei file sorgenti il proprio nome e numero di matricola come commento.

Quesito 1

Questo esercizio prevede l'uso di codice esistente da integrare e completare in modo da risolvere il problema proposto. Lo studente **deve utilizzare il codice proposto senza alcuna modifica** intervenendo solo nelle parti richieste ed integrandolo se necessario.

Perché l'esercizio sia sufficiente è necessario che il codice compili **senza errori**, il programma esegua senza **errori run-time**, e fornisca il risultato previsto.

```
#include <stdio.h>
#include <stdlib.h>
/*
La funzione legge un vettore di numeri interi minori di 100 fino all'inserimento di un valore negativo o il
raggiungimento del numero massimo di elementi n. La funzione restituisce il numero dei valori effettivamente letti
*/
int lettura_dati(int* v, int n) {
        for (int i = 0;i < n; ++i) {
                scanf("%d", &v[i]);
                if (v[i] < 0)
                        return i;
                if (v[i] >= 100)
                        --i;
        }
        return n;
}
/*
La funzione dato un valore reale e un vettore di ingresso che contiene valori maggiori zero restituisce un nuovo
vettore contenente tutti i valori pari presenti nel vettore di ingresso moltiplicati per il valore reale passato come
parametro.
Il vettore deve essere allocato all'interno della funzione della dimensione corretta
        la funzione restituisce (return) il vettore generato
        dim è la dimensione del vettore di ingresso
        vettoreIngresso è il vettore di ingresso
        *dim2 consente di restituire la dimensione del vettore
*/
float *prodotto_scalare_pari(int* vettoreIngresso, int dim, int *dim2) {
        // Il codice deve essere inserito dallo studente
}
/*
La funzione stampa su console il contenuto di un vettore di float
*/
void stampa( /* i parametri devono essere inseriti dallo studente */ ) {
        // Il codice deve essere inserito dallo studente
}
int main(void) {
        int *v1;                // vettore allocato dinamicamente
        float *v2;              // vettore allocato dinamicamente
        int dim;                // dimensione del vettore v1
        int dim2;               // dimensione del vettore v2
                // Scrivere il codice per leggere da tastiera il numero massimo di elementi inseribili nel vettore v1

        // Lettura da tastiera del primo vettore utilizzando la funzione lettura_dati
        dim = lettura_dati( /* lo studente inserisca i parametri corretti */);

        // Lo studente deve chiamare la funzione prodotto_scalare_pari salvando i valori nel vettore v2 allocato
        // all'interno della funzione.

        // lo studente, usando la funzione stampa, stampa il risultato contenuto in v2
        stampa(/* inserire i parametri*/);
        // disallocazione
}
```

I QUESITI SUCCESSIVI NON VERRANNO CORRETTI SE IL PROGRAMMA DEL QUESITO 1 NON FUNZIONA CORRETTAMENTE COME DESCRITTO ALL=INIZIO DELL'ESERCIZIO

Quesito 2

Si vuole realizzare un programma per la gestione delle code di attesa in supermercato per tre tipologie di merci:

- macelleria (tipo 0)
- pescheria (tipo 1)
- salumeria (tipo 2)

Ogni cliente si registra all'ingresso per una o più tipologia indicando il proprio nome ed eventualmente il diritto ad essere servito con priorità (valore booleano).

Le prenotazioni vengono raccolte in un vettore di code in base alla tipologia. Per ogni prenotazione è memorizzato il nome del cliente

Lo studente implementi un programma ANSI C dotato di un opportuno ***main*** in cui sia previsto un menu di scelta delle operazioni richieste quant'altro necessario a fare funzionare il programma comprese le eventuali operazioni di ingresso/uscita. Tutti i valori necessari al funzionamento devono essere passati utilizzando parametri, **non è permesso l'uso di variabili globali.**

Operazioni

1. funzione **inserimento_cliente** che riceve come parametro la lista di code e i dati della prenotazione e li inserisce nella coda appropriata. I dati devono essere acquisiti da console prima di chiamare la funzione;
2. funzione **prossimi_clienti** che restituisce in un vettore i tre clienti da servire (primi in coda) eliminandoli dalle rispettive code;
3. funzione **elenco_prioritari**, che data un tipo di coda (macelleria, pescheria, salumeria) cerca tutti i clienti prioritari e li restituisce in una coda;
4. funzione **attesa_prevista**, che dato il nome di un cliente, il tipo di coda restituisce e il tempo stimato di attesa in minuti (intero) per servire un cliente o -1 se il cliente non è presente. Il tempo necessario per gestire un cliente è passato come parametro ed è uguale per tutti i clienti presenti nella coda.

Quesito 3

Dato il valore -10 calcolare il valore codificato con 8 bit utilizzando la codifica in modulo e segno e in complemento a due

Modulo e Segno	Complemento a due

Quesito 4

Indicare due algoritmi di ricerca utilizzabili in un vettore ordinato specificando la loro complessità media. Descrivere almeno uno dei due algoritmi.

COGNOME E NOME	MATRICOLA	E
AULA	POSTAZIONE	

Istruzioni per lo svolgimento della prova

La prova consta di quattro quesiti, i primi due da svolgere al calcolatore il terzo e il quarto su carta.

La soluzione del quesito 1 in modo sufficiente (cioè il programma deve compilare ed eseguire correttamente) è condizione necessaria per il superamento della prova.

I file sorgenti (.c) relativi alla soluzione dei quesiti 1 e 2 devono essere consegnati su Studium nella sezione elaborati in un unico file compresso (preferibilmente .zip o .rar) contenente entrambi i due file. Inserire all'inizio dei file sorgenti il proprio nome e numero di matricola come commento.

Quesito 1

Questo esercizio prevede l'uso di codice esistente da integrare e completare in modo da risolvere il problema proposto. Lo studente **deve utilizzare il codice proposto senza alcuna modifica** intervenendo solo nelle parti richieste ed integrandolo se necessario.

Perché l'esercizio sia sufficiente è necessario che il codice compili **senza errori**, il programma esegua senza **errori run-time**, e fornisca il risultato previsto.

```
#include <stdio.h>
#include <stdlib.h>
/*
La funzione deve acquisire i dati da tastiera inserendoli in un vettore di dimensione n solo se i valori sono
superiori ad una soglia data. La funzione restituisce il numero dei valori effettivamente letti
*/
int lettura_dati(int* v, int n, int soglia) {
        for (int i = 0;i < n; ++i) {
                scanf("%d", &v[i]);
                if (v[i] < 0)
                        return i;
                if (v[i] > soglia)
                        --i;
        }
        return n;
}
/*
La funzione dati due vettori in ingresso v1 e v2 restituisce un vettore, allocato all'interno della funzione, che
contiene solo i valori pari presenti nel primo che sono presenti anche nel secondo vettore.
Il vettore deve essere allocato all'interno della funzione della dimensione corretta
        La funzione restituisce (return) il vettore generato
*/
int *intersezione ( /* il tipo ed il numero dei parametri è definito dallo studente */) {
        // Il codice deve essere inserito dallo studente
}
/*
La funzione stampa su console il contenuto di un vettore di interi
*/
void stampa(const int *v, int n) {
        // Il codice deve essere inserito dallo studente
}

int main(void) {
        int *v1;                // vettore allocato dinamicamente
        int *v2;                // vettore allocato dinamicamente
        int *v_risultato;       // vettore allocato dinamicamente
        int dim1, dim2;
        int dim_risultato;

        // Scrivere il codice necessario a leggere da tastiera del numero massimo di elementi inseribili
        // nel vettore v1 e nel vettore v2 (i valori potrebbero essere differenti)

        // Lettura da tastiera del primo vettore utilizzando la funzione lettura_dati
        dim1 = lettura_dati( /* lo studente inserisca i parametri corretti */);

        // Lettura da tastiera del secondo vettore utilizzando la funzione lettura_dati
        dim2 = lettura_dati( /* lo studente inserisca i parametri corretti */);

        // Lo studente deve chiamare la funzione intersezione salvando i valori nel vettore v2 allocato all'interno
        // della funzione.
        v_risultato = intersezione( /* lo studente deve inserire i parametri corretti */ );
```

```
        // lo studente, usando la funzione stampa, stampa il risultato contenuto in v_risultato
        stampa(/* inserire i parametri*/);

        // disallocazione
}
```

I QUESITI SUCCESSIVI NON VERRANNO CORRETTI SE IL PROGRAMMA DEL QUESITO 1 NON FUNZIONA CORRETTAMENTE COME DESCRITTO ALL=INIZIO DELL'ESERCIZIO

Quesito 2

Si vuole realizzare un programma per di una serie di partite di pallavolo. Ogni record contiene i seguenti campi:

- nome della prima squadra
- numero di set vinti dalla prima squadra
- nome della seconda squadra
- numero di set vinti dalla seconda squadra
- Data (nel formato AAAA-MM-GG, ad esempio 2019-07-03)

I dati devono essere memorizzati in una lista verificando che la partita non sia già presente.

Lo studente implementi un programma ANSI C dotato di un opportuno ***main*** in cui sia previsto un menu di scelta delle operazioni richieste e quant'altro necessario a fare funzionare il programma comprese le eventuali operazioni di ingresso/uscita. Tutti i valori necessari al funzionamento devono essere passati utilizzando parametri, **non è permesso l'uso di variabili globali.**

Operazioni

1. funzione **inserimento_partita** che riceve come parametro la lista di partite e i dati della partita e fornisce in uscita la lista aggiornata. In caso la partita sia già presente deve essere segnalato con un opportuno codice di errore. I dati devono essere acquisiti da console prima di chiamare la funzione;
2. funzione **calcola_punteggio** che, dato in ingresso il nome di una squadra restituisce il suo punteggio supponendo che ogni vittoria per 3-0 e 3-1 vale tre punti per la vincente e zero per la perdente, 3-2 vale due punti per la vincente e 1 punto per la perdente;
3. funzione **calcola_punteggio_tutte,** nell'ipotesi che il numero di squadre sia non superiore a 10, calcola il punteggio di tutte le squadre e lo restituisce in un vettore;
4. Funzione **elenco_partite**, funzione che restituisce in una lista ordinata rispetto alla data l'elenco delle partite

Quesito 3

Dato il valore 10010011 (8 bit) indicare il corrispondente valore decimale nel caso in cui si utilizzi la rappresentazione in modulo e segno e in Complemento a due.

Modulo e Segno	Complemento a due

Quesito 4

Descrivere l'algoritmo di inserimento in un albero binario di ricerca e applicarlo alla seguente sequenza di ingresso: 14, 8, 18, 6, 10, 16, 22. Indicare la profondità dell'albero risultante (profondità della radice = 0).

COGNOME E NOME	MATRICOLA	F
AULA	POSTAZIONE	

Istruzioni per lo svolgimento della prova

La prova consta di quattro quesiti, i primi due da svolgere al calcolatore il terzo e il quarto su carta.

La soluzione del quesito 1 in modo sufficiente (cioè il programma deve compilare ed eseguire correttamente) è condizione necessaria per il superamento della prova.

I file sorgenti (.c) relativi alla soluzione dei quesiti 1 e 2 devono essere consegnati su Studium nella sezione elaborati in un unico file compresso (preferibilmente .zip o .rar) contenente entrambi i due file. Inserire all'inizio dei file sorgenti il proprio nome e numero di matricola come commento.

Quesito 1

Questo esercizio prevede l'uso di codice esistente da integrare e completare in modo da risolvere il problema proposto. Lo studente **deve utilizzare il codice proposto senza alcuna modifica** intervenendo solo nelle parti richieste ed integrandolo se necessario.

Perché l'esercizio sia sufficiente è necessario che il codice compili **senza errori**, il programma esegua senza **errori run-time**, e fornisca il risultato previsto.

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

void LeggiVettore(int *v, int n)
{
  for (int i=0; i<n; i++) {
    printf("Inserisci elemento di indice %d: ", i);
    scanf("%d", &v[i]);
  }
}

/* Dato v1 di dimensione d1, restituisce un array di dimensione d2 che contiene gli elementi di v1 intervallati dalla
media dei valori contigui di v1. Per esempio, se v1 = {1,2,5,8} il vettore restituito sarà {1.0, 1.5, 2.0, 3.5, 5.0,
6.5, 8.0} */
float* Interpola(int* v1, int d1, int* d2) {...}

// Visualizza gli elementi di un array di float passati come parametro
void VisualizzaVettore(...) {...}

int main(void)
{
  int   *vett1;
  float *vett2;
  int dim1, dim2;

  printf("Inserisci la dimensione del vettore: ");
  scanf("%d", &dim1);

  vett1 = ...;  // Alloca un array di dim1 elementi interi
  LeggiVettore(...);

  vett2 = Interpola(...);

  VisualizzaVettore(...); // Visualizza vett2

  free(vett1);
  free(vett2);

  return 0;
}
```

I QUESITI SUCCESSIVI NON VERRANNO CORRETTI SE IL PROGRAMMA DEL QUESITO 1 NON FUNZIONA CORRETTAMENTE COME DESCRITTO ALL=INIZIO DELL'ESERCIZIO

Quesito 2

Una stazione meteo riceve le misure di temperatura e umidità di sensori dislocati in 5 diverse località. Ogni sensore comunica la temperatura media, l'umidità minima e massima giornaliera e la località in cui è avvenuta la misura. Lo studente implementi un programma ANSI C che consenta mantenere lo storico delle misurazioni giornaliere riportate dai vari sensori. Scrivere un opportuno ***main*** in cui sia previsto un menu di scelta delle operazioni richieste. Tutti i valori necessari al funzionamento devono essere passati utilizzando parametri, **non è permesso l'uso di variabili globali.**

1. Acquisizione da file lo storico delle misurazioni (guardare esempio in calce al quesito) e memorizzazione in un array di liste di misurazioni (una lista per ogni località).
2. Funzione che restituisce un vettore che contiene la temperatura massima registrata in ogni località.

3. Funzione che data una località ed una temperatura soglia (passate come parametro) restituisca una lista contenente tutte le misurazioni in cui la temperatura è superiore alla temperatura soglia.
4. Funzione che visualizzi per ogni località l'escursione di umidità massima.

Esempio di file

```
// TMedia, Umidità minima, Umidità massima, località
20 40 60 2
15 30 50 1
23 45 80 3
18 35 60 1
```

Quesito 3

Descrivere il ruolo dei bus in un sistema di elaborazione.

Quesito 4

Costruire l'albero binario di ricerca assumendo il seguente ordine di inserimento: 2, 9, 4, -5, 0, 7, 21, 28, 14. Visualizzare la sequenza dei nodi visitati se si effettua la visita differita (sinistra, destra, radice).