

7  
Programmazione I – Gruppo 2 – Prova di valutazione – 26/01/2022

Candidato: .....

Matricola: .....

Firma: .....

Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma.

Scrivere un programma C che:

- 1) Definisce un tipo *Stack* come un record che include un vettore di interi
- 2) Implementa le funzioni void *Push*(*Stack* \**S*, int *val*) e int *Pop*(*Stack* \**S*)
- 3) Legge da riga di comando una sequenza di numeri e usa lo stack per invertire la sequenza inserendola in un vettore *V*, che poi stampa a video.

Input e Output

Input: 1 12 3 4 5 6 7

Output:

La sequenza invertita è: 7 5 6 4 3 12 1

Candidato: .....

Matricola: ..... Firma: .....

Qs. 1 (b) – (3 punti): sia dato il vettore  $V = \{81, 3, 72, 20, 2, 1\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Bubble Sort.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	81	3	72	20	2	1
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>
double M[5]={0};

int main()
{
    int i=0;
    double *p, *q;
    int max = 5;

    for(i=1; i<5; i++)
        M[i] = max/i;

    p = M;
    q = &M[4];

    for(; p!=q; p++, q--)
        printf("if %f\n", *p, *q);

    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include <stdio.h>
int main(void){
    char int='a';
    char c="a", *pc=NULL;
    pc = &c;
    for(i<5; i++)
        pc = pc + 1;
    printf("&c\n", *pc);
    return void;
};
```

---



---



---



---



---



---



---



---



---



---



---



Candidato:

Matricola:

Firma:

**Istruzioni: LEGGERE ATTENTAMENTE E COMPILEARE PRIMA DI INIZIARE LA PROVA**

<b>Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti</b>	<b>Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.</b>	
<b>Per ciascun quesito riportare:</b> 1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false. Le domande per le quali non viene fornita alcuna risposta sono conteggiate come errori.  2) Solo ed esattamente quanto richiesto dalla traccia.	<b>Riservato alla commissione</b> <b>Voti parziali</b> <input type="checkbox"/> Qs. 1 a: ___ b: ___ <input type="checkbox"/> Qs. 2 ___ <input type="checkbox"/> Qs. 3 ___ <input type="checkbox"/> Qs. 4 ___ <input type="checkbox"/> Qs. 5 ___  <b>Voto Complessivo</b> <span style="font-size: 1.5em;">/30</span>	

**ESERCIZI**

**Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.**

N.	Domanda	V/F
1	Una istanza è la risoluzione di un problema su uno specifico insieme di dati	
2	L'analisi sintattica verifica la correttezza del significato delle istruzioni di un programma	
3	Il modello di Von Neumann prevede che solo i dati, ma non le istruzioni risiedano in memoria	
4	Il complemento a uno di un numero A ad n bit si calcola come $2^n - 1 - A$	
5	L'operatore <code>sizeof()</code> calcola il numero di bit occupato da un tipo di dato	
6	L'istruzione <code>int a=b=5;</code> dichiara due interi e li inizializza entrambi col valore 5	
7	L'operatore <code>~</code> (not) complementa tutti i bit di una variabile, tranne quello si segno	
8	L'istruzione <code>break</code> se usata fuori da un ciclo termina il programma	
9	Dati due vettori dichiarati come <code>int a[5], b[5];</code> l'assegnazione <code>a=b</code> è possibile perché hanno la stessa dimensione	
10	L'istruzione <code>int M[]/[3]={\{1,2,3\},\{4,5,6\},\{7,8,9\}};</code> è corretta perché <code>M</code> è un array di array	
11	La somma di due variabili char non può essere effettuata se rappresentano caratteri	
12	Il confronto di due record si effettua mediante l'operatore di uguaglianza <code>=</code>	
13	Le due forme <code>const int x</code> e <code>int const x</code> sono equivalenti	
14	Dato un vettore <code>int V[10];</code> ed un puntatore <code>int *p;</code> le due assegnazioni <code>p=V;</code> e <code>p=&amp;V[0];</code> sono equivalenti	
15	Per dimensioni dell'input n molto grandi il <i>Bubble Sort</i> è più efficiente del <i>Counting Sort</i>	
16	Uno stack implementato con un array ha capacità illimitata	
17	Una coda è sempre implementata con una lista doppiamente concatenata	
18	La ricorsione con stack esplicito richiede più memoria di quella implementata dallo stack di sistema	
19	La complessità di un problema corrisponde alla complessità del peggior algoritmo che lo risolve	
20	Il problema della Torre di Hanoi ha complessità $O(n^2 \log_2 n)$	

Candidato: .....

Matricola: .....

Firma: .....

**Qs. 1 (b) – (3 punti): sia dato il vettore  $V = \{81, 3, 72, 20, 2, 1\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Insertion Sort.**

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	81	3	72	20	2	1
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua					
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					

**Qs. 3 – (5 punti): si indichi l'output del seguente programma.**

```
#include <stdio.h>

struct parola {
    int lunghezza;
    char caratteri[8];
};

int main()
{
    int i=0;
    char *p;

    struct parola Testo = {5, "casa"};

    p = Testo.caratteri;
    for(i=Testo.lunghezza; Testo.caratteri[i]!='\0'; i++)
        Testo.caratteri[i] = *(p++);

    printf("%s\n", Testo.caratteri);

    return 0;
}
```

Output:

**Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.**

```
#include <stdio.h>
int main(int argc, char **argv){
static c = 0;
int (*fp) = 0;
scanf("ef", fp);
for(int i=fp; i<10; i++){
    if(1)
        continue(i=0);
}
return fp;
}
```

---



---



---



---



---



---



---



---



---



---



Candidato: .....

Matricola: ..... Firma: .....

Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:

- 1) Definisce un tipo *Stack* come un record che include un vettore di 100 interi
- 2) Implementa le funzioni

`void Push_vec(Stack *S, int v[], int n)`, la quale inserisce nello stack tutti i valori contenuti nel vettore *v*.

`int *Pop_vec(Stack *S, int n)`, la quale restituisce un vettore di interi contenente i primi *n* valori estratti dallo stack.

- 3) Legge un elenco di *n* interi da riga di comando, li inserisce nello stack, estraе *n/2* interi dallo stack e li stampa a video.

Input e Output

Input: 1 2 3 4 5 6 7 8 9 10

Output:

I valori estratti sono: 10 9 8 7 6

Candidato: .....

Matricola: ..... Firma: .....

**Istruzioni: LEGGERE ATTENTAMENTE E COMPILEARE PRIMA DI INIZIARE LA PROVA**

**Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti**

**Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.**

**Per ciascun quesito riportare:**

1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false. Le domande per le quali non viene fornita alcuna risposta sono conteggiate come errori.

2) Solo ed esattamente quanto richiesto dalla traccia.

**Riservato alla commissione**

**Voti parziali**  
 Qs. 1 a: \_\_\_ b: \_\_\_  
 Qs. 2 \_\_\_\_\_  
 Qs. 3 \_\_\_\_\_  
 Qs. 4 \_\_\_\_\_  
 Qs. 5 \_\_\_\_\_

**Voto Complessivo**

..... /30

---

ESERCIZI

---

**Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.**

N.	Domanda	V/F
1	Il codice binario di $221_{10}$ è $11011101_2$	
2	Una lista circolare ha un puntatore alla testa e uno alla coda della lista	
3	L'istruzione <i>else</i> si riferisce sempre all'istruzione <i>if</i> più vicina	
4	Le istruzioni <i>while(0){...}</i> e <i>for(;;){...}</i> sono equivalenti	
5	Uno stack implementato con un array ha capacità illimitata	
6	La funzione <i>strstr(...)</i> cerca una stringa in un'altra stringa	
7	La memoria allocata dalla funzione <i>calloc()</i> risiede nello stack di sistema	
8	Un puntatore a funzione non può essere dichiarato costante	
9	In un array di array la prima dimensione può essere omessa se viene inizializzato al momento della definizione	
10	Un puntatore a void non può essere dereferenziato	
11	La ricorsione con stack esplicito può implementare qualunque funzione	
12	Per gli interi negativi shift logico e shift aritmetico sono equivalenti	
13	Il tipo <i>char</i> occupa 1 byte su qualunque architettura (32 bit, 64 bit)	
14	La tabella ASCII contiene 256 simboli, quella estesa 512 simboli	
15	La notazione asintotica $\Omega$ , indica che una funzione limita superiormente un'altra funzione	
16	L'algoritmo <i>Insertion Sort()</i> ha complessità $O(n^2)$ in qualunque caso (migliore, medio, peggiore)	
17	Il passaggio di un array come parametro ad una funzione avviene sempre per puntatore	
18	La parola chiave <i>typedef</i> rimpiazza un tipo già esistente con quello appena definito	
19	Le istruzioni <i>x = ++x + 1</i> e <i>x = x++ + 1</i> sono equivalenti	
20	Una lista circolare può contenere solo un numero limitato di nodi	

**Programmazione I – Gruppo 2 – Prova di valutazione – 16/03/2022**

Candidato: .....

Matricola: ..... Firma: .....

Qs. 1 (b) – (3 punti): sia dato il vettore  $V = \{89, 1, 64, 11, 25, 40\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Selection Sort.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	89	1	64	11	25	40
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>

typedef union parola {
    int a;
    double b;
    char caratteri[8];
}parola;

int main(){
    char *p;
    parola P;

    p=P.caratteri;
    for(int i=7; i>=0; i--)
        *(p++) = 'z'-i;

    printf("%ld\n", sizeof(P));
    for(int i=0; i<8; i++)
        printf("%c ", P.caratteri[i]);

    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include <stdio.h>
int main(int argc, char *argv) {
static c = 0;
int (*fp) = 0;
scanf("%f", fp);
for(int i=fp; i<10; i++) {
    if(1)
        continue(i=0);
}
return fp; }
```

Candidato:

Firma:

Matricola:

Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

A

B

Scrivere un programma C che:

- 1) Definisce un tipo Digit come una lista, i cui nodi contengono il campo int bit
- 2) Implementa le funzioni Digit \*Dec2Dig(int val) che:
  - calcola il numero di bit n necessari a rappresentare val in binario;
  - Crea e restituisce la lista, i cui nodi contengono i bit dell'intero val
- 3) prende da linea di comando un valore intero e stampa i bit inseriti nella lista

Input e Output

Input: 26

Output:

I bit di 26 sono: 1->1->0->1->0->

di

/F

Nome:  
Cognome:

Punto:

(A)

**Istruzione: LEGGERE ATTENTAMENTE E COMPIERE PRIMA DI INIZIARE LA PROVA.**

**Servire in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti.**

**Servire in maniera legibile. Barrare nel rispondere soltanto i quesiti, a cui si intende rispondere.**

Per ciascun quesito riportare:

- 1) Una "V" o una "F", in corrispondenza delle risposte ritenute rispettivamente vere o false. Le domande per le quali non viene fornita alcuna risposta sono conteggiate come errori.

- 2) Solo ed esattamente quanta richiesto dalla traccia.

**Riservato alla commissione**

Voti parziali:

- Qs. 1. è: \_\_\_
- Qs. 2. \_\_\_
- Qs. 3. \_\_\_
- Qs. 4. \_\_\_
- Qs. 5. \_\_\_

Voto Complessivo

**/30****ESERCIZI**

Qs. 1 (a) - (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.

N.	Domanda	V/F
1	Il codice binario di $153_{10}$ è $10011001_2$	
2	Una lista circolare può contenere un nodo sentinella	
3	L'istruzione <code>else</code> si riferisce sempre all'istruzione <code>if</code> più esterna	
4	Se definita come <code>double D[10][5]</code> la variabile D è di tipo double **	
5	Il confronto di due stringhe non può essere effettuato mediante l'operatore ==	
6	Una coda può essere implementata con un array	
7	Una union e un record con un solo campo occupano la stessa quantità di memoria	
8	Il tipo <code>double</code> occupa 4 byte su una macchina a 32 bit	
9	Un puntatore a funzione può essere passato come parametro ad una funzione	
10	La funzione <code>malloc</code> restituisce un puntatore a intero	
11	La notazione asintotica O grande, indica che una funzione limita superiormente un'altra funzione	
12	La parola chiave <code>typedef</code> rimpiazza un tipo già esistente con quello appena definito	
13	Il passaggio di un array come parametro ad una funzione avviene sempre per puntatore	
14	Nei cicli for annidati, il ciclo interno controlla il ciclo esterno	
15	Nel caso migliore Bubble Sort ha complessità O(nlogn)	
16	La ricorsione sfrutta lo stack di attivazione del sistema	
17	Se si accede a un campo di un record r con l'operatore <code>r-&gt;</code> , r è un puntatore	
18	I puntatori hanno sempre la dimensione di una parola macchina	
19	Uno stack è una struttura ricorsiva di tipo FIFO	
20	L'inserimento di un nodo in una lista single-linked richiede due puntatori nella ricerca della posizione	



Candidate:

Matricola:

Firma:

Qs. 1 (b) - (3 punti): sia dato il vettore V={90, 5, 72, 8, 33, 1}. Si mostri lo stato del vettore ad ogni passo dell'algoritmo Insertion Sort.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	90	5	72	8	33	1
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua
11
12
13
14
15
16
17
18
19
20

Qs. 3 - (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>
int main()
{
    int i, j, s;
    int M[4][4]={{1,2,3,5,7},{0,2,4,6},{7,5,3,1},{6,4,2,0}};
    for(i=0; i<4; i++){
        s=0;
        for(j=0; j<4; j++)
            s += M[i][j];
        if(s>12)
            printf("%d %d\n", i, s);
    }
    return 0;
}
```

Output:

Qs. 4 - (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include "stdio.h"
int main(int argc, char *argv[]){
    int i=0, j;
    char C[]={"a","b","c","d","e"};
    for(i<5, i++){
        if(C[i]=='c')
            printf("%c ", C[i]+4);
    }
    return;
}
```

Candidato:

Matricola:

Firma:

Qs. 5 - (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:

- 1) Prende in input dalla linea di comando una parola
- 2) Definisce un nodo CharNode come un record che contiene un carattere e un puntatore al nodo successivo
- 3) Implementa la funzione Interco \*crea\_lista(char parola[])) che prende in input una parola e restituisce il puntatore alla testa di una lista di caratteri

Input e Output

Input: imprescindibile

La lista è: i->m->p->r->e->s->c->i->n->d->i->b->s->l->e->NULL.

Candidato: .....

Matricola: ..... Firma: .....

**Istruzioni: LEGGERE ATTENTAMENTE E COMPILARE PRIMA DI INIZIARE LA PROVA**

<p><b>Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti</b></p> <p>Per ciascun quesito riportare:</p> <p>1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false. Le domande per le quali non viene fornita alcuna risposta sono conteggiate come errori.</p> <p>2) Solo ed esattamente quanto richiesto dalla traccia.</p>	<p><b>Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.</b></p>	
<p><b>Riservato alla commissione</b></p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%; vertical-align: top;"> <b>Voti parziali</b>  <input type="checkbox"/> Qs. 1 a: ___ b: ___  <input type="checkbox"/> Qs. 2 _____  <input type="checkbox"/> Qs. 3 _____  <input type="checkbox"/> Qs. 4 _____  <input type="checkbox"/> Qs. 5 _____                 </td> <td style="width: 50%; vertical-align: top;"> <b>Voto Complessivo</b>                  ..... /30                 </td> </tr> </table>	<b>Voti parziali</b> <input type="checkbox"/> Qs. 1 a: ___ b: ___ <input type="checkbox"/> Qs. 2 _____ <input type="checkbox"/> Qs. 3 _____ <input type="checkbox"/> Qs. 4 _____ <input type="checkbox"/> Qs. 5 _____	<b>Voto Complessivo</b> ..... /30
<b>Voti parziali</b> <input type="checkbox"/> Qs. 1 a: ___ b: ___ <input type="checkbox"/> Qs. 2 _____ <input type="checkbox"/> Qs. 3 _____ <input type="checkbox"/> Qs. 4 _____ <input type="checkbox"/> Qs. 5 _____	<b>Voto Complessivo</b> ..... /30	

**ESERCIZI**

**Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.**

N.	Domanda	V/F
1	Nel caso migliore Bubble Sort ha complessità O(nlogn)	
2	La ricorsione sfrutta lo stack di attivazione del sistema	
3	L'istruzione <i>else</i> si riferisce sempre all'istruzione <i>if</i> più esterna	
4	Se definita come <i>double D[10][5]</i> la variabile D è di tipo double **	
5	La parola chiave <i>typedef</i> rimpiazza un tipo già esistente con quello appena definito	
6	Una lista circolare può contenere un nodo sentinella	
7	Uno stack è una struttura ricorsiva di tipo FIFO	
8	Il tipo double occupa 4 byte su una macchina a 32 bit	
9	Un puntatore a funzione può essere passato come parametro ad una funzione	
10	La funzione <i>malloc</i> restituisce un puntatore a intero	
11	Il codice binario di $153_{10}$ è $10011001_2$	
12	Una union e un record con un solo campo occupano la stessa quantità di memoria	
13	L'inserimento di un nodo in una lista single-linked richiede due puntatori nella ricerca della posizione	
14	Nei cicli for annidati, il ciclo interno controlla il ciclo esterno	
15	Il passaggio di un array come parametro ad una funzione avviene sempre per puntatore	
16	Il confronto di due stringhe non può essere effettuato mediante l'operatore ==	
17	Se si accede a un campo di un record r con l'operatore <i>-&gt;</i> , r è un puntatore	
18	I puntatori hanno sempre la dimensione di una parola macchina	
19	La notazione asintotica O grande, indica che una funzione limita superiormente un'altra funzione	
20	Una coda può essere implementata con un array	

Candidato: .....

Matricola: ..... Firma: .....

Qs. 1 (b) – (3 punti): sia dato il vettore  $V = \{78, 3, 22, 51, 1, 9\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Bubble Sort.

Iterazione	$V[0]$	$V[1]$	$V[2]$	$V[3]$	$V[4]$	$V[5]$
1	78	3	22	51	1	9
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>
int main()
{
    int i,j;
    char M[][]={{'a','b','c','d'},
                {'e','f','g','h'},
                {'i','j','k','l'},
                {'m','n','o','p'}};

    for(i=0; i<4; i++){
        for(j=0; j<4; j++)
            if(M[i][j]%2==0)
                printf("%c ", M[i][j]);
    }

    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include "stdio.h"
int main(int argc){
    int i,j;
    double M[]={1,2,3,4,5,6.0};
    while(i=0; i<6; i++){
        if(M[i][j]%2==0)
            printf("%c ", M[i][j]);
    }
    return -1;
}
```

---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---

Candidato: .....

Matricola: ..... Firma: .....  
Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:

- 1) Prende un vettore di n interi da linea di comando
- 2) Definisce un nodo *Elemento* come un record che contiene due interi pos e val, e un puntatore al nodo successivo
- 3) Implementa la funzione *Elemento \*crea\_vettore\_sparso(int vettore[], int len)* che crea una lista di nodi *Elemento*, ovvero scorre il vettore in input e se l'elemento in posizione i-esima del vettore è diverso da zero, crea un nodo, imposta il campo pos a i e il campo val a vettore[i] e lo inserisce nella lista.

Input e Output

Input: 20005000-8000-110003

Output:

Il vettore sparso è: (0, 2) -> (4, 5) -> (8, -8) -> (12, -11) -> (16, 3) -> NULL

A

Candidato: .....

Matricola: .....

Firma: .....

**Istruzioni: LEGGERE ATTENTAMENTE E COMPILARE PRIMA DI INIZIARE LA PROVA**

Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti

Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.

Per ciascun quesito riportare:	Riservato alla commissione
1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false. Le domande per le quali non viene fornita alcuna risposta sono conteggiate come errori. 2) Solo ed esattamente quanto richiesto dalla traccia.	<input type="checkbox"/> Qs. 1 a: ___ b: ___ <input type="checkbox"/> Qs. 2 _____ <input type="checkbox"/> Qs. 3 _____ <input type="checkbox"/> Qs. 4 _____ <input type="checkbox"/> Qs. 5 _____
	..... /30

ESERCIZI

Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.

N.	Domanda	V/F
1	Il compilatore ed il linker traducono il codice sorgente in un eseguibile	
2	Il bit più significativo è detto LSB	
3	La direttiva #define definisce costanti simboliche	
4	L'operatore -= sottrae l'r-value al l-value	
5	In C una stringa è una matrice di caratteri	
6	Un numero binario di 4n bit corrisponde ad un numero esadecinale di n cifre	
7	L'operazione di casting forza una conversione esplicita di tipo	
8	L'ordinamento interno è usato per insiemi di dati di piccola taglia	
9	Un vettore è una struttura dati ricorsiva	
10	Per un algoritmo la complessità di spazio è sempre trascurabile	
11	Uno stack implementato con un array ha capacità illimitata	
12	La complessità di tempo di un algoritmo è una stima asintotica	
13	Nel ciclo do{...}while(); il corpo viene eseguito almeno una volta	
14	int v[10] è un vettore di 11 interi	
15	La concatenazione di due stringhe si ottiene comando carattere per carattere	
16	La ricerca binaria in un vettore ordinato richiede O(logn) operazioni	
17	if(N=0){...} esegue sempre le istruzioni nelle parentesi graffe	
18	Il compilatore ottimizza il codice intermedio	
19	I vettori sono collezioni di variabili eterogenee	
20	Una union contiene sempre un solo dato, che però può essere di tipo diverso	

Candidato: \_\_\_\_\_

Monitoria: \_\_\_\_\_

Qs. 1 (b) - (5 punti): sia dato il vettore  $V = [32, 21, 33, 48, 13, 8]$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Insertion Sort.

Iterazione	$V[0]$	$V[1]$	$V[2]$	$V[3]$	$V[4]$	$V[5]$
1						
2	32					
3	21					
4						
5						
6						
7						
8						
9						
10						

Continua						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						

Qs. 3 - (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>
int main()
{
    int i, j;
    double x=1;
    int M[4][4] = { {65,'a', 65,'b'},
                    {65,'c', 65,'d', 65},
                    {65,'e', 65,'f'},
                    {65,'g', 65,'h', 65} };

    for (i = 0; i < 4; i++)
        x *= M[i][i];
    for (j = 0; j < 4; j++)
        if (x < 0)
            printf("in %d, M[%d][%d]\n",
                   i, j);
    return 0;
} //Nota: x/=65.
```

Output:

Qs. 4 - (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include <stdio.h>
int main(int argc, char *argv[])
{
    int i;
    float V[4][4] = { 1,2,3,4 };
    for (i = 0; i < 3; i++)
        if (V[i][i] > 2 == 0)
            printf("if %d ", V[i]);
}
return 0;
```

---



---



---



---



---



---



---



---

Candidato:

Firma:

A

Matricola:

Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che;

- 1) Prende in input dalla linea di comando una parola.
- 2) Definisce uno stack AStack a capacità limitata il cui array interno sia allocato dinamicamente con capacità pari alla lunghezza della parola in input.
- 3) Usa lo stack per invertire la parola.
- 4) Usa lo stack per verificare se la parola è palindroma.

Input e Output

Input: onorarono

La parola è palindroma e la sua inversione è onorarono.

Candidato: .....

Matricola: ..... Firma: .....

**Istruzioni: LEGGERE ATTENTAMENTE E COMPILEARE PRIMA DI INIZIARE LA PROVA**

<b>Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti</b>	<b>Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.</b>
<b>Riservato alla commissione</b>	
<b>Per ciascun quesito riportare:</b> 1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false. Le domande per le quali non viene fornita alcuna risposta sono conteggiate come errori.  2) Solo ed esattamente quanto richiesto dalla traccia.	<b>Voti parziali</b> <input type="checkbox"/> Qs. 1 a: ____ b: ____ <input type="checkbox"/> Qs. 2 _____ <input type="checkbox"/> Qs. 3 _____ <input type="checkbox"/> Qs. 4 _____ <input type="checkbox"/> Qs. 5 _____
	<b>/30</b>

---

ESERCIZI

---

Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.

N.	Domanda	V/F
1	Il compilatore ed il linker traducono il codice sorgente in un eseguibile	
2	La funzione <code>malloc</code> alloca memoria nello stack	
3	La ricorsione con stack esplicito è meno efficiente della ricorsione che fa uso dello stack di sistema	
4	La funzione <code>malloc</code> restituisce un puntatore a void *	
5	Le istruzioni <code>x = x + 1</code> , <code>x += 1</code> , <code>x++</code> e <code>++x</code> sono equivalenti	
6	Un numero binario di $4n$ bit corrisponde ad un numero esadecinale di $4n$ cifre	
7	L'algoritmo <i>Bubble Sort</i> ottimizzato ha complessità $O(n)$ nel caso migliore	
8	Nell'aritmetica dei puntatori, se $p$ è un puntatore ad intero, $p++$ è incrementato di 4 byte su una macchina a 32 bit	
9	Un vettore è una struttura dati non ricorsiva	
10	Un puntatore a funzione non può essere passato come parametro ad una funzione	
11	Uno stack implementato con un array ha capacità limitata	
12	La complessità di tempo di un algoritmo è rappresentata con un valore esatto	
13	Le istruzioni <code>while (i &lt; j) { ... }</code> e <code>for (i = j; i &lt; j) { ... }</code> sono equivalenti	
14	<code>int v[10]</code> è un vettore di 10 interi	
15	La concatenazione di due stringhe si ottiene interfolgiando carattere per carattere	
16	La ricerca binaria in un vettore ordinato richiede $O(\log n)$ operazioni	
17	<code>if(N=1){ ... }</code> esegue sempre le istruzioni nelle parentesi graffe	
18	Il linker ottimizza il codice intermedio	
19	I vettori sono collezioni di variabili omogenee	
20	Una union contiene sempre un solo dato, che però può essere di tipo diverso	

Candidato:

Matricola:

Firma:

Qs. 1 (b) – (3 punti): sia dato il vettore  $V = \{81, 3, 64, 23, 1, 2\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Insertion Sort.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	81	3	64	23	1	2
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua
11
12
13
14
15
16
17
18
19
20

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>

int main()
{
    int i=0, j=0;
    double s=0;
    int M[4][4] = { {1,-1, 1,-1},
                    {-1, 1,-1,1}};

    for (i = 0; i < 4; i++) {
        for (j = 0; j < 4; j++)
            if((i+j)%2)
                printf("%d \n", *(M[i]+j));
    }

    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include 'stdio.h'
int main(int argc, char *argv[])
{
    int i+j;
    float V[4][4] = { {1,2,3,4} };
    for (i = 0; i < 5; i++) {
        if (V[i] % 2 == 0)
            printf("%f ", V[i]);
    }
    return i;
}
```

Candidato: .....

Matricola: ..... Firma: .....

Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:

- 1) Prende da linea di comando 10 interi e li inserisce in un vettore
- 2) Implementa la funzione ricorsiva int Conta\_Occorrenze(int V[], int n, int val) che restituisce il numero di volte che il valore val compare nel vettore V.

Input e Output

Input: 5 8 6 12 9 5 2 5 1 8

Output:

Il valore 5 compare 3 volte

**Istruzioni: LEGGERE ATTENTAMENTE**

**Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti**

Per ciascun quesito riportare:

- 1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false, per ciascuna opzione relativa alle domande a risposta multipla.
- 2) Solo ed esattamente quanto richiesto dalla traccia.

**Scrivere in stampatello prima di inviare la prova**

**Sottostante i quesiti, a cui si intende rispondere**

**Riservato alla commissione**

- |   |                  |
|---|------------------|
| Voti parziali   | Voto Compositivo |
| <input type="checkbox"/> Qs. 1: <u>  </u> V<br><input type="checkbox"/> Qs. 2: <u>  </u> V<br><input type="checkbox"/> Qs. 3: <u>  </u> V<br><input type="checkbox"/> Qs. 4: <u>  </u> V<br><input type="checkbox"/> Qs. 5: <u>  </u> V | <b>130</b>       |

**ESERCIZI**

**Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.**

- |    |  |
|----|--|
| N. | <b>Domanda</b>   |
| 1  | Il codice binario di $114_{10}$ è $1101001_2$                              |
| 2  | Il linker ottimizza il codice oggetto                                      |
| 3  | Lo shift logico tiene conto del bit di segno                               |
| 4  | La sequenza di escape '\n' termina sempre una stringa                      |
| 5  | La funzione strcmp(a,b) restituisce 0 se le due stringhe a e b sono uguali |
| 6  | Una union occupa una quantità di memoria pari al suo tipo più piccolo      |
| 7  | L'ordinamento interno è usato per insiemi di dati di piccola taglia        |
| 8  | Una funzione ricorsiva non sempre può essere trasformata in una iterativa  |
| 9  | La complessità O() di tempo di un algoritmo è una stima per difetto        |
| 10 | Per un algoritmo la complessità di spazio è sempre trascurabile            |
| 11 | In una lista vuota il puntatore next della Testa è NULL                    |
| 12 | Il caso migliore per il BubbleSort è un vettore contrordinato              |
| 13 | Una variabile static viene distrutta a ogni chiamata di una funzione       |
| 14 | La funzione printf("%c", 'a') stampa il carattere a                        |
| 15 | L'operazione di push inserisce un nuovo elemento in una coda               |
| 16 | La funzione free libera memoria allocata dinamicamente                     |
| 17 | Un char corrisponde a 32 bit   |
| 18 | I numeri reali sono rappresentati in complemento a due                     |
| 19 | Una union occupa una quantità di memoria pari al suo tipo più piccolo      |
| 20 | L'istruzione continue interrompe un ciclo                                  |

Programmazione I – Gruppo 2 – Prova di valutazione – 11/01/2023

Candidato:

Matricola:

Qs. 1 (b) – (3 punti): sia dato il vettore  $V = [33, 21, 3, 1, 54, 10]$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo

Insertion Sort.

Firma:

Iterazione	$V[0]$	$V[1]$	$V[2]$	$V[3]$	$V[4]$	$V[5]$
1	33	21	3	1	54	10
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua
11
12
13
14
15
16
17
18
19
20

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>
int a = 1;
int main(int argc, char *argv[]) {
    int a = 2, b = 3;
    for (int a = 0; a < 3; a++)
        b = b + a;
    printf("a: %d, b: %d\n", a, b);
    printf("a: %d\n", (a++) - 1);
    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include <stdio>
int main(){
    char c = "dopo";
    int len = strlen(c);
    for(; len>0; len--)
        len -= (len--);
    printf("%s ", len);
    if (len = 0)
        continue(1);
    return &c;
}
```

Candidato:

Matricola:

Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:

- 1) Definisce un Punto come un record con i campi float x e y.
- 2) Implementa uno stack con capacità limitata di 10 punti, con le funzioni void Push(Stack \*S, Punto P) e Punto Pop(Stack \*S)
- 3) Inserisce in sequenza nello stack i punti P1(12,3), P2(14,4) e P3(7,9). Successivamente estrae un punto alla volta dallo stack e stampa le coordinate del punto estratto.

Input e Output

Input:

Output:

Primo punto P(7,9)

Secondo punto P(14,4)

Terzo punto P(12,3)

Candidato: .....  
Firma: .....  
Matricola: .....

**Istruzioni: LEGGERE ATTENTAMENTE E COMPILARE PRIMA DI INIZIARE LA PROVA**

Scrivere in stampatello Cognome, Nome e  
Matricola su ogni foglio. Usare per la prova  
esclusivamente i fogli forniti

Scrivere in maniera leggibile. Barrare nel riquadro  
sottostante i quesiti, a cui si intende rispondere.

**Riservato alla commissione**

**Voti parziali**

- Qs. 1 a: \_\_\_ b: \_\_\_
- Qs. 2 \_\_\_\_\_
- Qs. 3 \_\_\_\_\_
- Qs. 4 \_\_\_\_\_
- Qs. 5 \_\_\_\_\_

**Voto Complessivo**

...../30

**ESERCIZI**

Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.

N.	Domanda	V/F
1	Lo shift logico tiene conto del bit di segno	
2	L'ordinamento interno è usato per insiemi di dati di piccola taglia	
3	L'operazione di push inserisce un nuovo elemento in una coda	
4	L'istruzione continue interrompe un ciclo	
5	Un char corrisponde a 32 bit	
6	In una lista vuota il puntatore next della Testa è NULL	
7	Il caso migliore per il BubbleSort è un vettore contrordinato	
8	La funzione strcmp(a,b) restituisce 0 se le due stringhe a e b sono uguali	
9	Una funzione ricorsiva non sempre può essere trasformata in una iterativa	
10	Il linker ottimizza il codice oggetto	
11	I numeri reali sono rappresentati in complemento a due	
12	Per un algoritmo la complessità di spazio è sempre trascurabile	
13	Una union occupa una quantità di memoria pari al suo tipo più piccolo	
14	La funzione free libera memoria allocata dinamicamente	
15	Una union occupa una quantità di memoria pari al suo tipo più piccolo	
16	La sequenza di escape '\n' termina sempre una stringa	
17	Il codice binario di $114_{10}$ è $1101001_2$	
18	Una variabile static viene distrutta a ogni chiamata di una funzione	
19	La complessità O() di tempo di un algoritmo è una stima per difetto	
20	La funzione printf("%c", 'a') stampa il carattere a	

Programmazione I – Gruppo 2 – Prova di valutazione – 11/01/2023

Candidato: .....

Matricola: .....

Firma: .....

Qs. 1 (b) – (3 punti): sia dato il vettore  $V = \{12, 2, 35, 1, 4, 0\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Selection Sort.

Iterazione	$V[0]$	$V[1]$	$V[2]$	$V[3]$	$V[4]$	$V[5]$
1	12	2	35	1	4	0
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>
char parola[] = "dopo\0domani\0";
int main()
{
    int i = 0;
    char *p;

    p = parola;
    while (*p) {
        i++;
        p = p + 1;
    }

    printf("i = %d\n", i);

    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include [stdio.h]
int main(char argv[])
{
#define N 3;
float i=N;
N=5;
do(i<5){
    if(i-->0)
        continue(0);
while(&N);
return N; }
```



Programmazione I – Gruppo 2 – Prova di valutazione – 11/01/2023

B

Candidato: .....

Matricola: ..... Firma: .....

Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:

- 1) Prende dalla linea di comando due interi  $n$  e  $m$ , e  $n*m$  interi successivi. Alloca una matrice di interi  $M$  di dimensioni  $n \times m$  e la inizializza con gli interi letti in input.
- 2) Implementa la funzione `int mini_max_pari_dispari(int **M, int n, m, int *row, int *col)`, che individua la riga la cui somma degli elementi sia minima e la colonna la cui somma degli elementi sia massima.
- 3) Stampa nel main la riga e la colonna trovate dalla funzione.

Input e Output

Input: 3 4 12 33 22 11 3 67 1 23 76 34 12 1

Output:

La matrice è M:

012 033 022 011

003 067 001 023

076 034 012 001

La riga con somma minima è: 0

La colonna con somma massima è: 1

**Programmazione I – Gruppo 2 – Prova di valutazione – 09/02/2023**

Candidato: .....

Matricola: ..... Firma: .....

**Istruzioni: LEGGERE ATTENTAMENTE E COMPILARE PRIMA DI INIZIARE LA PROVA**

<b>Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti</b>  Per ciascun quesito riportare: 1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false, per ciascuna opzione relativa alle domande a risposta multipla. 2) Solo ed esattamente quanto richiesto dalla traccia.	<b>Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.</b>
<b>Riservato alla commissione</b>	<b>Voti parziali</b> <b>Voto Complessivo</b>
<input type="checkbox"/> Qs. 1 a: ___ b: ___ <input type="checkbox"/> Qs. 2 _____ <input type="checkbox"/> Qs. 3 _____ <input type="checkbox"/> Qs. 4 _____ <input type="checkbox"/> Qs. 5 _____	<span style="font-size: 1.5em;">.....</span> <b>/30</b>

**ESERCIZI**

Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato d ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.

N.	Domanda	V/F
1	Il codice binario di 18110 è 101101012	
2	Il linker genera il codice macchina di un programma	
3	Lo shift aritmetico tiene conto del bit di segno	
4	La funzione strcat cerca una sottostringa data in una stringa di input	
5	Una funzione $f(n) = \Omega(g(n))$ quando $g(n)$ la limita superiormente asintoticamente	
6	L'ordinamento in place non richiede ulteriori strutture di appoggio	
7	Il counting sort ha una elevata complessità di spazio	
8	In una coda la testa possiede un puntatore al primo e all'ultimo nodo della lista	
9	Nel caso medio la complessità asintotica del BubbleSort è di $O(2n)$	
10	La funzione malloc alloca memoria nello stack di sistema	
11	Le istruzioni while(1){} e for(;;){} sono equivalenti	
12	Una union occupa una quantità di memoria pari alla somma dei tipi dei suoi membri	
13	L'istruzione break torna direttamente all'inizio del ciclo per l'iterazione successiva	
14	La funzione scanf("%s", v) legge una stringa da linea di comando	
15	argv[1] è una stringa che contiene sempre il nome del programma	
	La ricerca binaria non si può applicare ad un vettore contrordinato	
	Un char corrisponde a 16 bit	
	Nei numeri rappresentati in complemento a due, lo zero ha due rappresentazioni	
	Per l'aritmetica dei puntatori il tipo del puntatore è ininfluente	
	Un puntatore a void non può essere dereferenziato	

Programmazione I – Gruppo 2 – Prova di valutazione – 09/02/2023

Candidato: .....

Matricola: .....

Qs. 1 (b) – (3 punti): sia dato il vettore  $V = \{81, 3, 72, 5, 25, 1\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Insertion Sort.

Iterazione	$V[0]$	$V[1]$	$V[2]$	$V[3]$	$V[4]$	$V[5]$
1	81	3	72	5	25	1
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>
int main()
{
    int i, j;
    int V[3][4] = {{1,2,3}, {5,6,7}, {11,12,13}};
    for (i = 0; i < 3; i++)
        for (j = 0; j < 4; j++)
            if ((i + j) % 2 == 0)
                continue;
            else
                printf("%d\n", V[i][j]);
    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include "stdio.h"
int main(int argc, char *argv[])
{
    int M[3] = {1, 2, 3}, i;
    M[1] = 0;
    for (i = 0; i < 3; i++)
        - 21
        printf(M[i], "%d ");
    0);
}
```

---



---



---



---



---



---



---



---



---



---

**Programmazione I – Gruppo 2 – Prova di valutazione – 09/02/2023**

Candidato: .....

Matricola: ..... Firma: .....

**Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma**

Scrivere un programma C che:

- 1) Definisce un tipo **digit** come un record due campi **int n** e **int \*d**
- 2) Prende un intero **val** da linea di comando
- 3) Implementa le funzioni **void Det2Dig(int val, digit \*dval)** che:
  - calcola il numero di bit **n** necessari a rappresentare **val** in binario;
  - alloca **d** come vettore di **n** interi
  - pone in **d** i bit della rappresentazione binaria di **val**.

**Input e Output**

Input: 57

Output:

La rappresentazione binaria di 57 è: 0 1 1 1 0 0 1

B

Candidato:

Matricola:

Firma:

**Istruzioni: LEGGERE ATTENTAMENTE E COMPIERE PRIMA DI INIZIARE LA PROVA**

Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti

Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.

Per ciascun quesito riportare:

1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false, per ciascuna opzione relativa alle domande a risposta multipla.

2) Solo ed esattamente quanto richiesto dalla traccia.

Riservato alla commissione

Voti parziali

- Qs. 1 a: \_\_\_ b: \_\_\_
- Qs. 2 \_\_\_\_\_
- Qs. 3 \_\_\_\_\_
- Qs. 4 \_\_\_\_\_
- Qs. 5 \_\_\_\_\_

Voto Complessivo

..... /30

— ESERCIZI —

Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.

N.	Domanda	V/F
1	Nel caso medio la complessità asintotica del BubbleSort è di $O(2n)$	
2	Un char corrisponde a 16 bit	
3	La funzione strcat cerca una sottostringa data in una stringa di input	
4	Per l'aritmetica dei puntatori il tipo del puntatore è influente	
5	L'istruzione break torna direttamente all'inizio del ciclo per l'iterazione successiva	
6	Le istruzioni while(1){} e for(;;{}) sono equivalenti	
7	Il counting sort ha una elevata complessità di spazio	
8	Una funzione $f(n)$ è $\Omega(g(n))$ quando $g(n)$ la limita superiormente asintoticamente	
9	Nei numeri rappresentati in complemento a due, lo zero ha due rappresentazioni	
10	Lo shift aritmetico tiene conto del bit di segno	
11	argv[1] è una stringa che contiene sempre il nome del programma	
12	La funzione scanf("%s", v) legge una stringa da linea di comando	
13	L'ordinamento in place non richiede ulteriori strutture di appoggio	
14	Un puntatore a void non può essere dereferenziato	
15	Il linker genera il codice macchina di un programma	
16	La funzione malloc alloca memoria nello stack di sistema	
17	Una union occupa una quantità di memoria pari alla somma dei tipi dei suoi membri	
18	In una coda la testa possiede un puntatore al primo e all'ultimo nodo della lista	
19	La ricerca binaria non si può applicare ad un vettore con ordinato	
20	Il codice binario di 18110 è 100101012	

(B)

Programmazione I – Gruppo 2 – Prova di valutazione – 09/02/2023

Candidato:

Matricola:

Qs. 1 (b) – (3 punti): sia dato il vettore  $V = [92, 32, 5, 11, 14, 30]$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Selection Sort.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	92	32	5	11	14	30
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>
int M[]={2,3,4,5,6,7};
int main()
{
    int i, a=1;
    printf("i=%d\n", a);

    {
        int a=0;
        for (i = 0; i < 6; i++)
            printf("%d\n", M[i+a]);
    }

    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include <stdio.h>
int main(int argc, char argv) {
    int i;
    float V[4][1]={{1},{2},{3},{4}};
    for (; ; ) {
        if (V[i][0] == 0)
            printf("%f ", V[i]);
    }
    return 6V;
}
```

---

---

---

---

---

---

---

---

---

---

---

B

Programmazione I – Gruppo 2 – Prova di valutazione – 09/02/2023

Candidato: .....

Matricola: .....

Firma: .....

Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:

- 1) Prende in input dalla linea di comando una parola
- 2) Definisce un nodo Carattere come un record che contiene un carattere e un puntatore al nodo successivo
- 3) Implementa la funzione Intero \*crea\_lista(char parola[]) che prende in input una parola e restituisce il puntatore alla testa di una lista di caratteri

Input e Output

Input: imprescindibile

La lista è: i->m->p->r->e->s->c->i->n->d->i->b->i->j->e->NULL

Candidato: .....

(A)

Matricola: .....

Firma: .....

Istruzioni: LEGGERE ATTENTAMENTE E COMPILARE PRIMA DI INIZIARE LA PROVA	
Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti	Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.
<b>Riservato alla commissione</b>	
Voti parziali	Voto Complessivo
<input type="checkbox"/> Qs. 1 a: ___ b: ___ <input type="checkbox"/> Qs. 2 _____ <input type="checkbox"/> Qs. 3 _____ <input type="checkbox"/> Qs. 4 _____ <input type="checkbox"/> Qs. 5 _____	...../30

ESERCIZI

Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando per ciascuna domanda una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa. Le domande lasciate in bianco sono considerate errate.

N.	Domanda	V/F
1	Il codice binario $10000100_2$ corrisponde al valore decimale $132_{10}$	
2	I campi di bit in un record possono contenere anche valori reali	
3	La ricerca binaria richiede un vettore aggiuntivo di appoggio per memorizzare i valori ordinati	
4	L'istruzione <i>break</i> in una serie di cicli innestati interrompe tutti i cicli	
5	L'istruzione <i>switch</i> verifica il valore di una espressione e opera a seconda dei diversi <i>case</i>	
6	L'aritmetica dei puntatori non permette il prodotto e la divisione fra puntatori	
7	La funzione <i>malloc</i> restituisce un puntatore a intero	
8	Un puntatore a void non può essere dereferenziato	
9	Il counting sort ha una complessità $O(n+k)$ dove $k$ è la lunghezza del vettore di appoggio	
10	I vettori paralleli sono coppie o insiemi di vettori indicizzati con un singolo indice	
11	Una variabile statica deve essere sempre inizializzata al momento della dichiarazione	
12	Le variabili globali sono allocate nello stack di sistema	
13	Una stringa vuota non ha il terminatore	
14	La tabella ASCII estesa ha 512 coppie carattere/codice	
15	Il linker elimina i commenti dal codice prima di compilare	
16	La notazione $O(n)$ si riferisce al numero esatto di operazioni svolto da un programma	
17	La versione ricorsiva dell'algoritmo di fibonacci ha una complessità maggiore della versione iterativa	
18	La torre di Hanoi è un problema la cui complessità è esponenziale	
19	In una coda con un singolo elemento i puntatori alla testa e alla coda hanno lo stesso valore	
20	Uno stack implementato con un array ha capacità illimitata	

Candidato: .....

Firma: .....

Matricola: .....  
Qs. 1 (b) – (3 punti): sia dato il vettore  $V = \{90, 5, 72, 3, 64, 1\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo  
Insertion Sort.

Iterazione	$V[0]$	$V[1]$	$V[2]$	$V[3]$	$V[4]$	$V[5]$
1	90	5	72	3	64	1
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>

int main()
{
    int i;
    int *p;
    int V[3][3] = {{1,2,3}, {5,6,7}, {11,12,13}};

    p = &(V[0][0]);

    for (i = 0; i < 3; i++){
        printf("%d -> ", V[i][2]);
        printf("%d\n", *p);
        ++p;
    }

    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include <stdio.h>
int main(int argc, char **argv){
static c = 0;
int (*fp) = 0;
scanf("%f", fp);
for(int i=fp; i<10; i++){
    if(1)
        continue(i=0);
}
return fp; }
```

Programmazione I – Gruppo 2 – Prova di valutazione – 10/03/2023

Candidato: .....

Matricola: ..... Firma: .....

Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:

- 1) Prende in input dalla linea di comando una parola
- 2) Implementa la funzione ricorsiva void *separa\_vocali(char V[], int n)*, che stampa prima tutte le vocali e poi tutte le consonanti

Input e Output

Input: vOlutAmEnte

Output:

La parola è: vOlutAmEnte

La partizione è: e E A u O v l t m n t

Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:

1) Prende dalla linea di comando 9 interi e un carattere c. Inserisce gli interi in una matrice M di dimensioni 3x3 allocata dinamicamente.

2) Implementa la funzione int \*RimpiazzaMaggiori(int \*\*M, char c), che calcola la media dei valori in M; e pone a 1 tutte le posizioni in M il cui valore è maggiore della media e a 0 quelle, il cui valore è minore della media. Infine, stampa a video una matrice di caratteri dove è presente il carattere c se il corrispondente valore in M è 1 e uno spazio altrimenti.

Input e Output |

Input: 12 1 15 5 10 3 11 0 9

La matrice binaria è:

1 0 1

0 1 0

1 0 1

Inserisci un carattere: A

La matrice di caratteri è:

A A

A

A A

Candidato: .....

Matricola: ..... Firma: .....

**Istruzioni: LEGGERE ATTENTAMENTE E COMPILEARE PRIMA DI INIZIARE LA PROVA**

<b>Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti</b>	<b>Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.</b>				
<p>Per ciascun quesito riportare:</p> <p>1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false, per ciascuna opzione relative alle domande a risposta multipla.</p> <p>2) Solo ed esattamente quanto richiesto dalla traccia.</p>	<p><b>Riservato alla commissione</b></p> <table border="1"> <tr> <td><b>Voti parziali</b></td> <td><b>Voto Complessivo</b></td> </tr> <tr> <td> <input type="checkbox"/> Qs. 1 a: _____ b: _____  <input type="checkbox"/> Qs. 2 _____  <input type="checkbox"/> Qs. 3 _____  <input type="checkbox"/> Qs. 4 _____  <input type="checkbox"/> Qs. 5 _____         </td> <td>..... <b>/30</b></td> </tr> </table>	<b>Voti parziali</b>	<b>Voto Complessivo</b>	<input type="checkbox"/> Qs. 1 a: _____ b: _____ <input type="checkbox"/> Qs. 2 _____ <input type="checkbox"/> Qs. 3 _____ <input type="checkbox"/> Qs. 4 _____ <input type="checkbox"/> Qs. 5 _____	..... <b>/30</b>
<b>Voti parziali</b>	<b>Voto Complessivo</b>				
<input type="checkbox"/> Qs. 1 a: _____ b: _____ <input type="checkbox"/> Qs. 2 _____ <input type="checkbox"/> Qs. 3 _____ <input type="checkbox"/> Qs. 4 _____ <input type="checkbox"/> Qs. 5 _____	..... <b>/30</b>				

**ESERCIZI**

Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando per ciascuna domanda una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa. Le domande lasciate in bianco sono considerate errate.

N.	Domanda	V/F
1	In un sistema numerico posizionale il valore di una cifra dipende dalla posizione e non dalla base	
2	Le variabili globali sono allocate nell'heap	
3	Per un problema con complessità $\Omega(n)$ non esiste un algoritmo con tempo sublineare	
4	L'istruzione <i>break</i> in una serie di cicli innestati interrompe il ciclo più esterno	
5	L'algoritmo iterativo per la sequenza di Fibonacci è più efficiente di quella ricorsiva non ottimizzata	
6	Il C fornisce l'operatore <i>sizeof</i> per determinare la dimensione di una struttura	
7	Il C consente loop nidificati, in cui un loop è inserito all'interno di un altro loop	
8	In C, l'operatore <i>++</i> ha una precedenza maggiore dell'operatore <i>*</i>	
9	L'istruzione <i>switch</i> può essere utilizzata solo con numeri interi.	
10	Il C supporta sia il passaggio dei parametri per valore che per riferimento.	
11	In una coda è possibile accedere direttamente agli elementi in posizioni arbitrarie senza seguirne l'ordine	
12	Le code in C possono essere implementate utilizzando uno stack	
13	In una funzione ricorsiva non necessariamente bisogna raggiungere il caso base	
14	L'algoritmo di ordinamento Selection Sort ha una complessità di tempo di $O(n \log n)$ nel caso peggiore	
15	L'allocazione dinamica della memoria in C viene eseguita utilizzando l'operatore <i>*</i> prima del puntatore	
16	In C è necessario liberare manualmente la memoria allocata dinamicamente per evitare memory leaks	
17	In C, una funzione può avere uno o più valori di ritorno	
18	Le funzioni in C possono richiamare se stesse all'interno del loro corpo	
19	In C, i parametri vengono sempre passati per valore alle funzioni	
20	In C, è possibile passare un array come parametro a una funzione senza specificarne la dimensione	

Candidato: .....

Matricola: ..... Firma: .....

**Qs. 1 (b) – (3 punti): sia dato il vettore  $V=\{43, 90, 1, 27, 33, 2\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Selection Sort.**

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	43	90	1	27	33	2
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						

**Qs. 3 – (5 punti): si indichi l'output del seguente programma.**

```
#include <stdio.h>

int main()
{
    int i,j;
    int *p;
    int V[3][3] = {{1,2,3}, {5,6,7}, {11,12,13}};

    p = &(V[0][1]);
    for (i = 0; i < 3; i++){
        --p;
        for (j = 0; j < 3; j++){
            printf("%d\n", *p);
            ++p;
        }
    }
    return 0;
}
```

**Output:**

**Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.**

```
#include 'stdio.h'
int M[3][]={{1,2,3},{4,5,6},{7,8,9}};
int main(char argc, int *argv[]){
    int i=0,j=i;
    for(, i<3, i++)
        do(j>0){
            if(i%0)
                printf(k, "%d");
        }while();
}
```

---



---



---



---



---



---



---



---



---



---

Candidato: .....

Matricola: .....

Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:

- 1) Prende una parola da linea di comando
- 2) Definisce un nodo Elemento come un record che contiene un intero pos e un carattere c, e un puntatore al nodo successivo
- 3) Implementa la funzione Elemento \*crea\_vettore\_sparsa(char vettore[], int len) che crea una lista di nodi Elemento, ovvero scorre il vettore in input e se l'elemento in posizione i-esima del vettore è una vocale, crea un nodo, imposta il campo pos a i e il campo c a vettore[i] e lo inserisce nella lista.

Input e Output

Input: Indissolubilmente

Output:

Il vettore sparso è: (0,i)->(3,i)->(6,o)->(8,u)->(10,i)->(13,e)->(16,e)->NULL

Candidato: .....

Firma: .....

Matricola: .....

**Istruzioni: LEGGERE ATTENTAMENTE E COMPILARE PRIMA DI INIZIARE LA PROVA**

Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti.

Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.

Per ciascun quesito riportare:

- 1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false, per ciascuna opzione relativa alle domande a risposta multipla.

2) Solo ed esattamente quanto richiesto dalla traccia.

Riservato alla commissione.

Voti parziali  
 Qs. 1: a: \_\_\_ b: \_\_\_  
 Qs. 2: \_\_\_  
 Qs. 3: \_\_\_  
 Qs. 4: \_\_\_  
 Qs. 5: \_\_\_

Voto Complessivo:

/30

**ESERCIZI**

Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando per ciascuna domanda una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa. Le domande lasciate in bianco sono considerate errate.

N.	Domanda	V/F
1	In una coda è possibile accedere direttamente agli elementi in posizioni arbitrarie senza seguirne l'ordine	
2	Il C consente loop nidificati, in cui un loop è inserito all'interno di un altro loop	
3	In una funzione ricorsiva non necessariamente bisogna raggiungere il caso base	
4	L'algoritmo di ordinamento Selection Sort ha una complessità di tempo di $O(n \log n)$ nel caso peggiore	
5	L'istruzione switch può essere utilizzata solo con numeri interi	
6	In C è necessario liberare manualmente la memoria allocata dinamicamente per evitare memory leaks	
7	In C, una funzione può avere uno o più valori di ritorno	
8	Le funzioni in C possono richiamarsi se stesse all'interno del loro corpo	
9	In C, i parametri vengono sempre passati per valore alle funzioni	
10	In C, è possibile passare un array come parametro a una funzione senza specificarne la dimensione	
11	L'istruzione break in una serie di cicli innestati interrompe tutti i cicli	
12	Il C non implementa il passaggio dei parametri per riferimento.	
13	L'allocazione dinamica della memoria in C utilizza la memoria dello stack	
14	In un sistema numerico posizionale il valore di una cifra dipende dalla posizione e dalla base	
15	In C una coda non può essere implementata con un array	
16	Il C fornisce l'operatore sizeof per determinare la dimensione di una struttura	
17	In C, l'operatore ++ ha una precedenza minore dell'operatore *	
18	Le variabili globali sono allocate nell'heap	
19	Per un problema con complessità $\Omega(n)$ non esiste un algoritmo con tempo sublineare	
20	L'algoritmo iterativo per la sequenza di Fibonacci è meno efficiente di quella ricorsiva ottimizzata	

Programmazione I – Gruppo 2 – Prova di valutazione – 13/06/2023

Candidato: .....

A

Matricola: .....

Firma: .....

Qs. 1 (b) – (3 punti): sia dato il vettore  $V = [88, 4, 62, 8, 1, 12]$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Insertion Sort.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	88	4	62	8	1	12
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua
11
12
13
14
15
16
17
18
19
20

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>
int main()
{
    int i,j;
    char *p;
    char V[3][3] = { {'0','1','2'}, { '3','4','5'}, { '6','7','8'} };
    p = &(V[0][0]);
    for (i = 0; i < 3; i++)
        p = p + 1;
        for (j = 0; j < 3; j++)
            printf("%c\n", *p+j);
    }
    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include <stdio.h>
int main(int argc, char **argv){
    int i=0;
    int *pc=0;
    pc = &i;
    for(i=0; i<5; i++)
        *pc = i;
    printf("%d\n", *pc);
    return 0;
}
```

---

---

---

---

---

Candidato: .....

Matricola: ..... Firma: .....  
Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:

- 1) Definisce una struttura Stack implementata con una lista singolarmente concatenata, in cui ciascun nodo contiene un valore intero e il puntatore al nodo successivo
- 2) Prende in input dalla linea di comando un valore int n
- 3) Implementa la funzione Stack \*crea\_stack\_fibonacci(int n) che prende in input un valore intero n e restituisce uno stack in cui sono inseriti i primi n numeri di fibonacci
- 4) Stampa il contenuto dello stack nella funzione main

Input e Output

Input: 9

Output:

Il Contenuto dello stack è: 1 1 2 3 5 8 13 21 34

Candidato: .....

Matricola: .....

Firma: .....

**Istruzioni: LEGGERE ATTENTAMENTE E COMPILARE PRIMA DI INIZIARE LA PROVA**

Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti

Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.

Per ciascun quesito riportare:

- 1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false. Le domande per le quali non viene fornita alcuna risposta sono conteggiate come errori.

- 2) Solo ed esattamente quanto richiesto dalla traccia.

**Riservato alla commissione**

**Voti parziali**

- Qs. 1 a: \_\_\_ b: \_\_\_
- Qs. 2 \_\_\_\_\_
- Qs. 3 \_\_\_\_\_
- Qs. 4 \_\_\_\_\_
- Qs. 5 \_\_\_\_\_

**Voto Complessivo**

...../30

**ESERCIZI**

Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.

N.	Domanda	V/F
1	La rappresentazione binaria dell'intero decimale 25 è 00011001	
2	Il Linguaggio C supporta nativamente la gestione delle liste doppiamente concatenate	
3	L'operatore di dereferenziazione * in C viene utilizzato per accedere al valore puntato da un puntatore	
4	L'operazione di inserimento in una lista singolarmente concatenata avviene sempre all'inizio della lista	
5	L'operatore di incremento postfisso x++ restituisce il valore di x dopo essere stato incrementato	
6	La complessità di tempo dell'algoritmo di ricerca binaria nel caso peggiore è O(log n)	
7	Un puntatore a void può essere assegnato a un puntatore di qualsiasi tipo in C	
8	Una struttura ricorsiva in C è una struttura che contiene come membro se stessa come semplice variabile	
9	Un puntatore a una struttura in C consente di accedere ai campi della struttura tramite l'operatore ->	
10	Il Linguaggio C consente la dichiarazione di variabili di tipo booleano	
11	Un array multidimensionale è una collezione di elementi dello stesso tipo disposti su una singola dimensione	
12	La ricorsione è un concetto che si riferisce alla chiamata di una funzione all'interno di se stessa	
13	Un record in C è una struttura di dati che può contenere variabili di diversi tipi	
14	La sequenza di Fibonacci è una sequenza di numeri in cui ogni numero è il massimo dei due precedenti	
15	Il Linguaggio C fornisce la libreria string.h che contiene funzioni per la manipolazione di stringhe	
16	L'operatore sizeof restituisce la dimensione in byte di una variabile di un determinato tipo	
17	Il passaggio di un array come parametro a una funzione richiede sempre la specifica della sua dimensione	
18	L'operatore di confronto == può essere utilizzato per confrontare due stringhe in C	
19	L'algoritmo di ordinamento Bubble Sort è un algoritmo stabile	
20	L'algoritmo di ordinamento Selection Sort è un algoritmo in-place, ovvero non richiede memoria aggiuntiva	

## Programmazione I – Gruppo 2 – Prova di valutazione – 07/07/2023

A

Candidato: .....

Matricola: ..... Firma: .....

Qs. 1 (b) – (3 punti): sia dato il vettore  $V = \{17, 5, 12, 8, 20, 3\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Bubble Sort.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	17	5	12	8	20	3
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua					
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>

typedef struct { int x; int y; } Point;
int main() {
    int i, j;
    Point A[2][2] = { {{1, 2}, {3, 4}}, {{5, 6}, {7, 8}} };

    Point *ptr = &A[0][0];
    for (i = 1; i > 0; i--) {
        for (j = 1; j > 0; j--) {
            printf("%d, %d ", (ptr+i*2+j)->x, (ptr+i*2+j)->y);
        }
    }
    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include [stdio.h]
int main(void){
    char c* = NULL;
    c = "test\0";
    for(int i=0; c[i]!='\0'; i++)
        printf('%c ', c[i]);
    free(c);
    c=c-1;
    i=*c;
    return (int) (*c); }
```

**Programmazione I – Gruppo 2 – Prova di valutazione – 07/07/2023**

Candidato: .....

Matricola: ..... Firma: .....

**Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma**

**Scrivere un programma C che:**

- 1) Rappresenta un treno come una lista di 11 nodi (ogni nodo è un record e rappresenta una carrozza), in cui un array di 100 elementi che rappresentano i posti (0 libero, 1 occupato)**
- 2) Implementa le funzioni int prenota\_posto(Treno T, int carrozza), trova la prima posizione libera nella carrozza corrispondente, la segna come occupata e restituisce la posizione come output.**
- 3) Implementa la funzione main(), definisce un treno come T, inizializza l'occupazione dei posti in modo casuale. Inoltre, prende da linea di comando il numero della carrozza in cui prenotare e stampa il posto prenotato nella carrozza.**

**Input e Output**

**Input: 9**

**Posto prenotato nella carrozza 9: posto 2**

Candidato: .....

Matricola: .....

Firma: .....

**Istruzioni: LEGGERE ATTENTAMENTE E COMPILARE PRIMA DI INIZIARE LA PROVA**

Scrivere in stampatello Cognome, Nome e Matricola su ogni foglio. Usare per la prova esclusivamente i fogli forniti

Scrivere in maniera leggibile. Barrare nel riquadro sottostante i quesiti, a cui si intende rispondere.

Per ciascun quesito riportare:

- 1) Una 'V' o una 'F', in corrispondenza delle risposte ritenute rispettivamente vere o false. Le domande per le quali non viene fornita alcuna risposta sono conteggiate come errori.

- 2) Solo ed esattamente quanto richiesto dalla traccia.

**Riservato alla commissione**

**Voti parziali**

- Qs. 1 a: \_\_\_ b: \_\_\_
- Qs. 2 \_\_\_\_\_
- Qs. 3 \_\_\_\_\_
- Qs. 4 \_\_\_\_\_
- Qs. 5 \_\_\_\_\_

**Voto Complessivo**

...../30

**ESERCIZI**

Qs. 1 (a) – (5 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.

N.	Domanda	V/F
1	La rappresentazione binaria dell'intero decimale 25 è 00011001	
2	Il Linguaggio C supporta nativamente la gestione delle liste doppiamente concatenate	
3	L'operatore di dereferenziazione * in C viene utilizzato per accedere al valore puntato da un puntatore	
4	L'operazione di inserimento in una lista singolarmente concatenata avviene sempre all'inizio della lista	
5	L'operatore di incremento postfisso x++ restituisce il valore di x dopo essere stato incrementato	
6	La complessità di tempo dell'algoritmo di ricerca binaria nel caso peggiore è O(log n)	
7	Un puntatore a void può essere assegnato a un puntatore di qualsiasi tipo in C	
8	Una struttura ricorsiva in C è una struttura che contiene come membro se stessa come semplice variabile	
9	Un puntatore a una struttura in C consente di accedere ai campi della struttura tramite l'operatore ->	
10	Il Linguaggio C consente la dichiarazione di variabili di tipo booleano	
11	Un array multidimensionale è una collezione di elementi dello stesso tipo disposti su una singola dimensione	
12	La ricorsione è un concetto che si riferisce alla chiamata di una funzione all'interno di se stessa	
13	Un record in C è una struttura di dati che può contenere variabili di diversi tipi	
14	La sequenza di Fibonacci è una sequenza di numeri in cui ogni numero è il massimo dei due precedenti	
15	Il Linguaggio C fornisce la libreria string.h che contiene funzioni per la manipolazione di stringhe	
16	L'operatore sizeof restituisce la dimensione in byte di una variabile di un determinato tipo	
17	Il passaggio di un array come parametro a una funzione richiede sempre la specifica della sua dimensione	
18	L'operatore di confronto == può essere utilizzato per confrontare due stringhe in C	
19	L'algoritmo di ordinamento Bubble Sort è un algoritmo stabile	
20	L'algoritmo di ordinamento Selection Sort è un algoritmo in-place, ovvero non richiede memoria aggiuntiva	

Candidate: .....

Matricola: ..... Firma: .....  
Qs. 1 (b) – (3 punti): sia dato il vettore  $V = \{9, 4, 12, 2, 7, 5\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo  
Insertion Sort.

Iterazione	$V[0]$	$V[1]$	$V[2]$	$V[3]$	$V[4]$	$V[5]$
1	9	4	12	2	7	5
2						
3						
4						
5						
6						
7						
8						
9						
10						

Continua						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						

Qs. 3 – (5 punti): si indichi l'output del seguente programma.

```
#include <stdio.h>

int sum(int a, int b) { return a + b; }
int subtract(int a, int b) { return a - b; }

int main() {
    int i, j;
    int (*operation[2])(int, int) = {sum, subtract};

    for (i = 1; i < 3; i++) {
        for (j = 1; j < 3; j++) {
            int result = operation[(i - 1) % 3](i, j);
            printf("%d ", result);
        }
        printf("\n");
    }
    return 0;
}
```

Output:

Qs. 4 – (5 punti): si indichino gli errori, qualora presenti, nelle seguenti istruzioni.

```
#include <stdio.h>
int main(int argc, char argv) {
    int i;
    float V[4][1]={{1},{2},{3},{4}};
    for ( ; ; ) [
        if (V[i][0] == 0)
            printf("%f ", V[i]);
    ]
    return &V;
```

---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---

Nome	Cognome
Giulio	Giuliano
20	08/08/2023
08/08/2023	08/08/2023
08/08/2023	08/08/2023

Informatica I – Gruppo 2 – Prova di valutazione – 07/07/2023

B

Candidato:

Matricola: \_\_\_\_\_ Firma: \_\_\_\_\_  
Qs. 5 – (12 punti): Si implementi in linguaggio C il seguente programma

Scrivere un programma C che:  
2) Definisce una lista doppiamente concatenata Lista\_DL, i cui nodi contengono un campo dato di tipo intero  
3) Implementa la funzione void Ordina\_Lista(Lista\_DL L), che ordina la lista L mediante l'algoritmo bubble sort applicato alla lista.  
4) Implementa la funzione main, che prende un elenco di numeri interi da linea di comando, li inserisce in una lista di tipo Lista\_DL e poi ordina la lista con la funzione Ordina\_Lista. Inoltre stampa la lista prima e dopo l'ordinamento.

Input e Output

Input: 12 1 34 6 20

Output:

Lista prima dell'ordinamento: 12 1 34 6 20  
Lista dopo l'ordinamento: 1 6 12 20 34

Qs. 1 - (5 + 3 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.

- 1) Il codice binario di  $104_{10}$  è  $1101001_2$
- 2) La ricerca binaria è applicabile anche su vettori non ordinati in  $O(n)$
- 3) Il linker compila il codice sorgente
- 4) La funzione `puchar()` inserisce un solo carattere nello standard output
- 5) Lo shift aritmetico tiene conto del bit di segno
- 6) Il tipo int occupa 4 byte su una macchina a 32 bit
- 7) `for(;;){...}` non esegue mai il corpo delle istruzioni
- 8) La funzione `strlen()` concatena due stringhe
- 9) In C si possono annidare più cicli do-while
- 10) Il token `piano_forte` è un identificatore valido
- 11) `if(S){...}` eseguito sempre il corpo delle funzioni
- 12) La sequenza di escape '\n' termina sempre una stringa
- 13) Nei cicli for annidati, il ciclo esterno controlla il ciclo interno
- 14) La funzione `strcmp(a,b)` restituisce 0 se le due stringhe a e b sono uguali
- 15) I record sono collezioni di variabili eterogenee
- 16) I campi di bit in un record possono essere di tipo float
- 17) `int M[10][5]` è una matrice di interi con 5 righe e 10 colonne
- 18) Una union occupa una quantità di memoria pari al suo tipo più piccolo
- 19) La funzione `malloc` alloca sempre memoria contigua
- 20) Dato un puntatore ad un record `r`, `(*r).campo` e `r->campo` sono equivalenti

Sia dato il vettore  $V = \{60, 23, 31, 1, 4, 32\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo Selection Sort.

Iterazione	$V[0]$	$V[1]$	$V[2]$	$V[3]$	$V[4]$	$V[5]$
1	60	23	31	1	4	32
2						
3						

↑    ↓    1 di 1    +    ?    🔍



LG

F5      F6      F7      F8      F9      F10      F11      F12      Stamp Resist      Back Space  
(      )      =      ?      ^      ←      Ins  
8      9      0      ,      1

**Qs. 2 – (5 punti): si indichi l'output del seguente programma, così come apparirebbe sul monitor di un computer, ovvero tenendo conto anche della formattazione.**

```
#include <stdio.h>

int main()
{
    int i, j;
    double s = 1;
    int M[][4] = {{0,'a',0,'b'},
                  {'c',0,'d',0},
                  {0,'e',0,'f'},
                  {'g',0,'h',0} };

    for (i = 0; i < 4; i++) {
        s = -s;
        for (j = 0; j < 4; j++)
            if (s > 0) {
                printf("%d %d\n", i, j);
                break;
            }
    }
    return 0;
}
```

**Qs. 3 – (5 punti): si individuino gli errori nel seguente programma. Per le righe, in cui è presente un errore, si scriva sulla linea corrispondente l'errore individuato. Per le righe, in cui non sono presenti errori, si scriva a fianco “Corretto”.**

```
1 #include "stdio.h"
2 int main(char argc, int *argv[]) {
3     int i = 0, j = 0.5;
4     double i = 1;
5     int M[][][4] = { {0,'a',0,'b'} } ;
6     while (i = 0; i < 4; i++) {
7         scanf(&j, "%d");
8         if (j < 0)
9             break(-1);
10        return &i;
}
```

**Qs. 4 – (12 punti):**

Scrivere un programma C che:

- 1) Prende da linea di comando una parola
- 2) Implementa la funzione `char *estrai_doppie(char parola[])` che
  - a) alloca dinamicamente un vettore di caratteri, inserisce nel vettore solo le lettere doppie, e lo restituisce
  - b) nel main stampa il contenuto del vettore restituito dalla funzione `estrai_doppie`

**Input e Output**

Input: appiattire

Output:

Le doppie nella parola appiattire sono: pt

**Qs. 1 – (5 + 3 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.**

- 1) Nei cicli for annidati, il ciclo interno controlla il ciclo esterno
- 2) La funzione `strcmp(a,b)` restituisce 1 se le due stringhe a e b sono uguali
- 3) `if(N=0){...}` esegue sempre le istruzioni nelle parentesi graffe
- 4) La funzione `tolower()` trasforma un carattere in maiuscolo
- 5) Il nome di un vettore è un puntatore al suo ultimo elemento
- 6) Se si accede a un campo di un record r con l'operatore `->`, r è un puntatore
- 7) `int v[10]` è un vettore di 11 interi
- 8) Una union contiene sempre un solo dato, che però può essere di tipo diverso
- 9) I vettori sono collezioni di variabili eterogenee
- 10) I campi di bit in un record non possono contenere valori decimali
- 11) Nel ciclo `do{...}while();` il corpo viene eseguito almeno una volta
- 12) La concatenazione di due stringhe si ottiene comando carattere per carattere
- 13) Per i dati signed shift logico e aritmetico sono equivalenti
- 14) Il tipo `char` occupa 2 byte
- 15) In C non si possono annidare più cicli while
- 16) Il token *piano-forte* è un identificatore valido
- 17) Il Bubble sort ha sempre complessità O(n)
- 18) La ricerca binaria in un vettore ordinato richiede O(log n) operazioni
- 19) La ricorsione sfrutta lo stack di attivazione del sistema
- 20) La funzione `getchar()` prende un solo carattere dallo standard input

Sia dato il vettore  $V=\{8, 53, 1, 61, 4, 10\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo *Bubble Sort*.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	8	53	1	61	4	10
2						
3						

**Qs. 2 – (5 punti): si indichi l'output del seguente programma, così come apparirebbe sul monitor di un computer, ovvero tenendo conto anche della formattazione.**

```
#include <stdio.h>

int M[3][3]={{1,2,3},{4,5,6},{7,8,9}};

int main()
{
    int i=0,j=0;

    for(i=0; i<3; i++)
        for(j=2; j>=0; j--){
            if(i==j)
                continue;
            printf("%d ", M[i][j]);
        }

    return 0;
}
```

**Qs. 3 – (5 punti): si individuino gli errori nel seguente programma. Per le righe, in cui è presente un errore, si scriva sulla linea corrispondente l'errore individuato. Per le righe, in cui non sono presenti errori, si scriva a fianco “Corretto”.**

```
1 #include 'stdio.h'  
2 int M[3][]={{1,2,3},{4,5,6},{7,8,9}};  
3 int main(char argc, int *argv[]) {  
4     int i=0,j=i;  
5     for(, i<3, i++)  
6         do(j>0){  
7             if(i%0)  
8                 printf(k, "%d");  
9         }while();  
10 }
```

**Qs. 4 – (12 punti):**

Scrivere un programma C che:

- 1) Prende una parola dalla linea di comando
- 2) Implementa la funzione `char *consonanti_lower(char parola[])` che restituisce una stringa allocata dinamicamente nella funzione, in cui le consonanti sono minuscole e le vocali sono maiuscole.

**Input e Output**

Input: DiSsImuLazioNE

Output:

La nuova parola è: dIsslmUlAzlOnE

**Qs. 1 – (5 + 3 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.**

- 1) Il SelectionSort ha complessità  $O(n\log n)$
- 2) L'ordinamento interno è usato per insiemi di dati di piccola taglia
- 3) Il caso migliore per il BubbleSort è un vettore contrordinato
- 4) Nel caso migliore BubbleSort ottimizzato è  $O(n)$
- 5) Il CountingSort ha complessità  $O(n)$
- 6) Un vettore è una struttura dati ricorsiva
- 7) Gli elementi di una lista sono record
- 8) Le liste sono strutture concatenate
- 9) I nodi di una lista bidirezionale non hanno un puntatore al predecessore
- 10) In una lista vuota il puntatore next della Testa è NULL
- 11) La ricerca in una lista bidirezionale con  $n$  nodi ha complessità  $O(\log n)$
- 12) Una coda è una struttura dati con politica LIFO
- 13) Una coda implementata con liste ha capacità limitata
- 14) Le principali operazioni su uno stack sono Push e Pop
- 15) Uno stack implementato con un array ha capacità illimitata
- 16) La ricorsione definisce una funzione mediante se stessa
- 17) Una funzione ricorsiva non può essere trasformata in una iterativa
- 18) La ricorsione sfrutta lo stack dei record di attivazione delle funzioni
- 19) La complessità di tempo di un algoritmo è una stima asintotica
- 20) Per un algoritmo la complessità di spazio è sempre trascurabile

Sia dato il vettore  $V = \{12, 44, 3, 1, 8, 32\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo *Insertion Sort*.

Iterazione	$V[0]$	$V[1]$	$V[2]$	$V[3]$	$V[4]$	$V[5]$
1	12	44	3	1	8	32

**Qs. 2 – (5 punti): si indichi l'output del seguente programma, così come apparirebbe sul monitor di un computer, ovvero tenendo conto anche della formattazione.**

```
#include <stdio.h>

int V[]={1, 2, 3, 4, 5, 6, 7, 8};

int main()
{
    int i, ris=0, *p;

    p = &V[3];
    for(i=*p; i<8; i++)
        ris += V[i];

    printf("ris = %d\n", ris);

    return 0;
}
```

**Qs. 3 – (5 punti): si individuino gli errori nel seguente programma. Per le righe, in cui è presente un errore, si scriva sulla linea corrispondente l'errore individuato. Per le righe, in cui non sono presenti errori, si scriva a fianco “Corretto”.**

```
1 #include <stdio.h>
2 int main(int argc, char argv) {
3     int i;
4     float V[4][1]={{1},{2},{3},{4}};
5     for (; ; ) [
6         if (V[i][0] == 0)
7             printf("%f ", V[i]);
8     ]
9     return &V;
10 }
```

**Qs. 4 – (12 punti):**

Scrivere un programma C che:

- 1) Definisce un tipo *Stack* come un record che include un vettore di interi
- 2) Implementa le funzioni void *Push*(*Stack* \**S*, int *val*) e int *Pop*(*Stack* \**S*)
- 3) Legge da riga di comando un intero *m* e alloca dinamicamente il vettore dello stack con dimensione *m*. Legge da riga di comando *m* valori interi e li inserisce nello stack con la funzione *Push*. Stampa a video i valori estratti dallo stack con la funzione *Pop* finché lo stack non è vuoto.

**Input e Output**

Input: 6 1 2 3 4 5 6

Output:

Il contenuto dello stack è: 6 5 4 3 2 1

**Qs. 1 – (5 + 3 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.**

- 1) Il *Bubble Sort* ha complessità  $O(2n)$
- 2) L’istruzione *continue* interrompe un ciclo
- 3) Una coda ha una politica di accesso di tipo FIFO
- 4) La funzione *free* libera memoria allocata dinamicamente
- 5) La *ricorsione* fa uso dello stack di sistema
- 6) I numeri reali sono rappresentati in complemento a due
- 7) Un ciclo *for* non può essere sempre trasformato in un ciclo *while*
- 8) L’operazione di *push* inserisce un nuovo elemento in una coda
- 9) In una lista circolare il nodo sentinella è sempre al centro della lista
- 10) In una lista bidirezionale bisogna conservare un puntatore al predecessore
- 11) La funzione *strlen* restituisce la lunghezza di una stringa
- 12) Un *char* corrisponde a 8 bit
- 13) Una variabile *static* viene distrutta a ogni chiamata di una funzione
- 14) Un record è una collezione omogenea di variabili
- 15) Una *union* ha la dimensione del tipo più piccolo che contiene
- 16) Un ciclo *do-while* viene eseguito almeno una volta
- 17) Il ciclo *for(;;)* equivale al ciclo *while(1)*
- 18) Lo *scope* di una variabile è la sua dimensione in memoria
- 19) La funzione *printf("%c", 'a')* stampa il carattere *a*
- 20) Il *counting sort* ha una elevata complessità di spazio

Sia dato il vettore  $V=\{8, 72, 3, 12, 44, 10\}$ . Si mostri lo stato del vettore ad ogni passo dell’algoritmo *Selection Sort*.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	8	72	3	12	44	10
2						
3						
4						
5						
6						
7						

**Qs. 2 – (5 punti): si indichi l'output del seguente programma, così come apparirebbe sul monitor di un computer, ovvero tenendo conto anche della formattazione.**

```
#include <stdio.h>

char parola[] = "dopo\0domani\0";

int main()
{
    int i = 0;
    char *p;

    p = parola;
    while (*p) {
        i++;
        p = p + 1;
    }

    printf("i = %d\n", i);

    return 0;
}
```

---

**Qs. 3 – (5 punti): si individuino gli errori nel seguente programma. Per le righe, in cui è presente un errore, si scriva sulla linea corrispondente l'errore individuato. Per le righe, in cui non sono presenti errori, si scriva a fianco “Corretto”.**

```
1 #include <stdio>
2 int main(){
3     char c = "dopo";
4     int len = strlen(c);
5     for(; len>0; len--)
6         len -= (len--);
7     printf("%s ", len);
8     if (len = 0)
9         continue(1);
10    return &c; }
```

**Qs. 4 – (12 punti):**

Scrivere un programma C che:

- 1) Prende due interi M ed N da linea di comando e alloca una matrice di *char* di M righe e N colonne
- 2) Inserisce nella posizione i,j della matrice il valore (65+i+j)
- 3) Implementa la funzione *void stampa\_a\_scacchi(char \*\*Matrice, int M, int N)* che stampa il valore della matrice quando la riga o la colonna è dispari e uno spazio altrimenti.

**Input e Output**

Input: 5 7

Output:

La matrice di interi M è:

B	D	F	
B	D	F	H
D	F	H	
D	F	H	J
F	H	J	

**Qs. 1 – (5 + 3 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.**

- 1) Il prototipo della funzione `calloc` è `int *calloc(int, int)`
- 2) La funzione `strlen` conta il numero di caratteri in una stringa prima dello '\n'
- 3) La *ricorsione* si basa sul principio di induzione
- 4) L'istruzione `for(;;)` non viene mai eseguita
- 5) Una espressione è considerata *vera* se il suo valore è maggiore di zero
- 6) La parola chiave `#define` definisce un nuovo tipo
- 7) Una variabile `static` mantiene il suo valore anche dopo che la funzione termina
- 8) Una dichiarazione `extern` non alloca memoria per una variabile statica
- 9) I record possono essere annidati dichiarando una *struct* all'interno di un'altra *struct*
- 10) Le istruzioni in un blocco `do-while()` vengono eseguite almeno una volta
- 11) Nel costrutto `switch()-case`, l'istruzione `break` nel caso `default` non è necessario
- 12) L'espressione `c = a | b`, pone in `c` l'and bit a bit fra le variabili `a` e `b`
- 13) La sequenza di bit  $11011111_2$  corrisponde alla sequenza esadecimale 0xDF
- 14) Il nome di un array è il puntatore al primo elemento dell'array
- 15) La variabile `int (*fp)(char)` è un puntatore a funzione che prende in input un `char` e restituisce un `int`
- 16) Il passaggio di un array ad una funzione è sempre per valore
- 17) La serie di Fibonacci è un problema con complessità  $O(2^n)$
- 18) Una funzione  $\Theta(n)$  è sia  $O(n)$  che  $\Omega(n)$
- 19) La soluzione iterativa della torre di Hanoi ha costo  $O(n)$
- 20) Se `p` è puntatore costante di tipo `int`, l'istruzione `*p=5` genera un errore

Sia dato il vettore  $V=\{98, 2, 53, 12, 24, 10\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo *Bubble Sort*.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	98	2	53	12	24	10
2						
3						
4						
5						
6						
7						

**Qs. 2 – (5 punti): si indichi l'output del seguente programma, così come apparirebbe sul monitor di un computer, ovvero tenendo conto anche della formattazione.**

```
#include <stdio.h>

int V[] = { 1, 8, 9, 12 };

int main()
{
    int i = 0;

    for (i = 0; i < 4; i++)
        switch (V[i] % 3) {
            case 0:
                printf("%d ", 0);
                break;
            case 1:
                printf("%d ", 1);
            case 2:
                printf("%d ", 2);
                break;
        }

    return 0;
}
```

---

**Qs. 3 – (5 punti): si individuino gli errori nel seguente programma. Per le righe, in cui è presente un errore, si scriva sulla linea corrispondente l'errore individuato. Per le righe, in cui non sono presenti errori, si scriva a fianco “Corretto”.**

```
1 #include [stdio.h]
2 int main(void) {
3     char c* = NULL;
4     c = "test\0";
5     for(int i=0; c[i]!='\0'; i++)
6         printf('%c ', c[i]);
7     free(c);
8     c=c-1;
9     i=*c;
10    return (int) (*c); }
```

**Qs. 4 – (12 punti):**

Scrivere un programma C che:

- 1) Definisce un tipo *digit* come un record due campi *int n* e *int \*d*
- 2) Implementa le funzioni void Det2Dig(int val, digit \*dval) che:
  - calcola il numero di bit *n* necessari a rappresentare *val* in binario;
  - alloca *d* come vettore di *n* interi
  - pone in *d* i bit della rappresentazione binaria di *val*.

Input e Output

Input: 57

Output:

La rappresentazione binaria di 57 è: 0 1 1 1 0 0 1

**Qs. 1 – (5 + 3 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.**

- 1) Il tipo long double ha precisione doppia rispetto ad un tipo double
- 2) L'istruzione for(;;) implementa un ciclo infinito
- 3) La funzione `calloc` restituisce un puntatore a intero
- 4) La funzione `strcmp` confronta i primi n caratteri di due stringhe
- 5) La parola "casa" rappresentata come stringa occupa 5 caratteri
- 6) La soluzione ricorsiva ottimizzata per il calcolo della sequenza di Fibonacci ha costo O(n)
- 7) La notazione  $g(x)=\Omega(f(x))$  indica un limite asintotico inferiore per  $g(x)$
- 8) Uno stack implementato con array allocato dinamicamente ha capacità illimitata
- 9) I nodi di una lista sono implementati con strutture ricorsive
- 10) Non è possibile dichiarare variabili di tipo `void`
- 11) Un puntatore NULL ha valore 0
- 12) Il nome M di una matrice int M[5][8] è di tipo int \*\*
- 13) Non è possibile fare il cast di una variabile double al tipo meno preciso int
- 14) In una lista circolare l'ultimo nodo punta a se stesso
- 15) In una coda implementata con array l'operazione di `enqueue` costa O(n)
- 16) La ricorsione con stack esplicito diminuisce la complessità asintotica di un algoritmo
- 17) La soluzione iterativa della Torre di Hanoi ha complessità asintotica  $O(n^2)$
- 18) L'operatore % non è applicabile a variabili di tipo double
- 19) L'operatore ++ è la contrazione dell'operatore += 1
- 20) Incrementare un puntatore di 1 significa sempre spostarlo in avanti di 4 byte

Sia dato il vettore  $V=\{42, 2, 15, 1, 14, 0\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo *Insertion Sort*.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1	42	2	15	1	14	0
2						
3						
4						
5						
6						
7						

**Qs. 2 – (5 punti): si indichi l'output del seguente programma, così come apparirebbe sul monitor di un computer, ovvero tenendo conto anche della formattazione.**

```
#include <stdio.h>
int main() {
    int M[3][3] = { {1,2,3},{4,5,6} };
    for (int i = 0, j = 0; i < 3; i++) {
        for (; j < 3; j++)
            break;
        printf("%d\n", M[i][j]);
    }
    return 0;
}
```

---

**Qs. 3 – (5 punti): si individuino gli errori nel seguente programma. Per le righe, in cui è presente un errore, si scriva sulla linea corrispondente l'errore individuato. Per le righe, in cui non sono presenti errori, si scriva a fianco “Corretto”.**

```
1 #include <stdio.h>
2 int main(int argc, char **argv) {
3     static c = 0;
4     int (*fp) = 0;
5     scanf("%f", fp);
6     for(int i=fp; i<10; i++) {
7         if(1)
8             continue(i=0);
9     }
10    return fp; }
```

**Qs. 4 – (12 punti):**

Scrivere un programma C che:

- 1) Prende dalla linea di comando una un vettore di n interi
- 2) Implementa la funzione ricorsiva void stampa\_pari\_dispari(int V[], int n), che stampa prima tutti i valori pari e poi tutti i valori dispari

Input e Output

Input: 12 3 4 6 8 1 9 2

Output:

La sequenza è: 2 8 6 4 12 3 1 9

**Qs. 1 – (5 + 3 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.**

- 1) Il compilatore ed il linker traducono il codice sorgente in un eseguibile
- 2) Il bit più significativo è detto *LSB*
- 3) Nel sistema binario il valore di una cifra dipende dalla sua posizione
- 4) Con  $n$  bit si rappresentano  $2^n - 1$  valori
- 5) La direttiva `#define` definisce costanti simboliche
- 6) `sizeof(char)` restituisce 1 su qualsiasi PC
- 7) La direttiva `#define` definisce costanti simboliche
- 8) L'operatore `--` sottrae l'*r-value* al *l-value*
- 9) La funzione `strlen` calcola la lunghezza di una stringa
- 10) In C una stringa è una matrice di caratteri
- 11) Un numero binario di  $4n$  bit corrisponde ad un numero esadecimale di  $n$  cifre
- 12) Il complemento a 1 di un numero A è  $B = 2^n - 1 + A$
- 13) Nel complemento a 2 lo zero ha una sola rappresentazione
- 14) In un ciclo `for` l'istruzione `continue` interrompe il ciclo
- 15) Il token `piano_forte` non è un identificatore valido
- 16) Il campo di visibilità di una variabile è detto *scope* della variabile
- 17) L'operazione di casting forza una conversione esplicita di tipo
- 18) Un costrutto `if/else` si può sempre riscrivere con un costrutto `switch/case/default`
- 19) La tabella ASCII comprende 256 simboli
- 20) Per copiare due record, si deve copiare ciascun membro

Sia dato il vettore  $V = \{40, 33, 1, 21, 4, 50\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo *Selection Sort*.

Iterazione	$V[0]$	$V[1]$	$V[2]$	$V[3]$	$V[4]$	$V[5]$
1	40	33	1	21	4	50
2						
3						

**Qs. 2 – (5 punti): si indichi l'output del seguente programma, così come apparirebbe sul monitor di un computer, ovvero tenendo conto anche della formattazione.**

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

char str[] = "Programmazione I";

int main()
{
    int i, n;

    n = strlen(str);

    for(i=0; i<n; i++) {
        if(i%2)
            continue;
        printf("%c ", str[i]);
    }

    return 0;
}
```

**Qs. 3 – (5 punti): si individuino gli errori nel seguente programma. Per le righe, in cui è presente un errore, si scriva sulla linea corrispondente l'errore individuato. Per le righe, in cui non sono presenti errori, si scriva a fianco “Corretto”.**

```
1 #include >stdio.h<
2 int main(double argc, char argv) {
3     struct {double a; float b;}p;
4     p.a = &p.b;
5     for (i = 0; i < 5; i++) {
6         if (i % p.a == 0)
7             printf("%f ", p.a);
8     }
9     return p.a;
10 }
```

**Qs. 4 – (12 punti):**

Scrivere un programma C che:

- 1) Prende in input dalla linea di comando una parola
- 2) Definisce un nodo *Carattere* come un record che contiene un carattere e un puntatore al nodo successivo
- 3) Implementa la funzione `int *crea_Lista(char parola[])` che prende in input una parola e restituisce il puntatore alla testa di una lista di caratteri

Input e Output

Input: imprescindibile

La lista è: i->m->p->r->e->s->c->i->n->d->i->b->i->l->e->NULL

**Qs. 1 – (5 + 3 punti): si risponda alle seguenti domande riportando su un foglio bianco il numero della domanda e al lato di ciascuna numero una V se si considera la risposta vera o una F se si ritiene che la risposta sia falsa.**

- 1) Il complemento a 2 di un numero A è  $B = 2^n - 1 + A$
- 2) Il Selection Sort ha complessità  $O(n\log n)$
- 3) Un vettore è una struttura dati ricorsiva
- 4) Una coda è una struttura dati con politica FIFO
- 5) La direttiva #define definisce costanti simboliche
- 6) La complessità di un algoritmo è una stima esatta del tempo di esecuzione
- 7) Una coda a capacità illimitata è implementata con liste
- 8) I nodi di una lista sono strutture ricorsive
- 9) La ricorsione sfrutta lo stack dei record di attivazione delle funzioni
- 10) Per un algoritmo la complessità di spazio è sempre trascurabile
- 11) int v[3][3] è un vettore di 3 vettori di 3 interi
- 12) Nei cicli while annidati, il ciclo interno controlla il ciclo esterno
- 13) Il preprocessore ottimizza il codice intermedio
- 14) Una union contiene sempre un puntatore all'elemento successivo
- 15) Lo shift logico tiene conto del bit di segno
- 16) La funzione `gets()` prende una intera stringa dallo standard input
- 17) L'istruzione `typedef` sostituisce un vecchio tipo con un nuovo tipo definito dall'utente
- 18) Il confronto fra due record in C si effettua confrontando i rispettivi campi
- 19) In C non si possono annidare più cicli `do-while`
- 20) Dato un puntatore ad un record r, `(*r).campo` e `r->campo` sono equivalenti

Sia dato il vettore  $V=\{81, 11, 35, 15, 4, 1\}$ . Si mostri lo stato del vettore ad ogni passo dell'algoritmo *Bubble Sort*.

Iterazione	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
1						
2						
3						
4						
5						
6						
7						
...						

**Qs. 2 – (5 punti): si indichi l'output del seguente programma, così come apparirebbe sul monitor di un computer, ovvero tenendo conto anche della formattazione.**

```
#include <stdio.h>
#include <ctype.h>
union { int a; long b; double f; }U;
int main() {
    U.f = 2.0;
    for (U.a = 0; U.a < 4; U.a++)
        printf("%d ", U.a);
    printf("%f\n", U.f = 3.5);
    return 0;
}
```

---

**Qs. 3 – (5 punti): si individuino gli errori nel seguente programma. Per le righe, in cui è presente un errore, si scriva sulla linea corrispondente l'errore individuato. Per le righe, in cui non sono presenti errori, si scriva a fianco “Corretto”.**

```
1 #include [stdio.h]
2 int main(char argv[]){  
3 #define N 3;  
4 float i=N;  
5 N=5;  
6 do(i<5){  
7     if(i-->0)  
8         continue(0);  
9     while(&N);  
10    return N; }
```

**Qs. 4 – (12 punti):**

Scrivere un programma C che:

- 1) Prende in input dalla linea di comando una stringa
- 2) Definisce un nodo *Carattere* come un record che contiene un carattere e un puntatore al nodo successivo
- 3) Implementa la funzione *Carattere \*CreaLista(char vettore[])* che prende in input una stringa e restituisce il puntatore alla testa di una lista di caratteri.

**Nota: i caratteri nella lista devono avere lo stesso ordine che hanno nella stringa**

Input e Output

Input: parolaio

La lista è: p->a->r->o->l->a->i->o->