

Nome e Cognome: \_\_\_\_\_

Matricola: \_\_\_\_\_

1. 5 punti Cosa stampa il seguente frammento di codice?

```

1  int a = 0123 ^ 0x056;
2  double b = 2.59;
3
4  printf ("%d\n", a);
5
6  while ((++a || a++) ? a--=1 : 0) {
7      if (!(a-- && --a ))
8          break;
9      else {
10         printf ("%d\n", a);
11     }
12 }
13
14 a<=a, a+=b, a++;
15 printf("a: %d\n", a);
16

```

5  
3  
1  
-1  
a: 4

2. 6 punti Elencare le conversioni di tipo implicita (... da ... a). Scrivere cosa viene stampato a schermo sapendo che:  $UCHAR\_MAX = 255$ ,  $'a' = 97$ .

```

1  double fun (float a) {
2      char b = ('x' * 3) - 'g';
3      return (a / b);
4  }
5
6  int main (void) {
7      unsigned int a = 'g' - 3UL;
8      float b = fun(a);
9      unsigned char c = -(int) (b+53);
10     printf("c: %c, %d\n", c, c);
11     return 0;
12 }
13

```

linea 7: 'g' convertito da int a unsigned long int  
 linea 7: il valore dopo l'uguale è convertito da unsigned long int ad unsigned int  
 linea 8: parametro "a" di fun convertito da unsigned int a float  
 linea 2: il valore dopo l'uguale è convertito da int a char  
 linea 3: "b" è convertito da char a float per la divisione  
 linea 3: il risultato della divisione è convertito da float a double  
 linea 8: il valore di ritorno è convertito da double a float  
 linea 9: 53 è convertito da int a float  
 linea 9: il valore dopo l'uguale è convertito da int (dopo la conversione esplicita) a unsigned char

A schermo viene stampato "c: g, 103" perchè:  
 $c = (UCHAR\_MAX + 1) - 153 = 103 = 'g'$  in ASCII

3. 6 punti Data la seguente *struct*, scrivere la definizione di una funzione di nome *ritorna\_dispari* che prende come parametro una lista (*lista\_input*) e ritorna un'altra lista (*lista\_output*, creata nella funzione) che contiene, nello stesso ordine della lista passata, solamente gli elementi in posizione *dispari* (se presenti). Se la lista originale è **5-2-9**, la lista ritornata sarà **5-9**.

```

1  typedef struct node Node;
2
3  struct node {
4      int info;
5      struct node* pNext;
6  };
7

```

Guarda soluzione in fondo al compito

Nome e Cognome: \_\_\_\_\_

Matricola: \_\_\_\_\_

4. **7 punti** Dire quali compilazioni provocano errore a causa del linker (e perchè):

- 1) gcc -o write write.c
- 2) gcc -c main.c
- 3) gcc -o main main.c
- 4) gcc -o execute main.c write.c

In caso il punto 4) ritorni un errore, descrivere come può essere corretto. Infine, **dopo la correzione** eventualmente applicata, elencare tutte le definizioni, dichiarazioni e tipologie di linkage, presenti in ogni file, per *count*, *i*, *a*, e *mywrite*. Cosa stampa il programma?

**main.c**

```
extern void mywrite(int *count);
extern int count;
int i;
int i = 1;

int main(void) {
    do {
        mywrite(&count);
    } while(count <= 0);
}
```

**write.c**

```
#include <stdio.h>

static int count = -3;

void mywrite(int *a) {
    static int count = 2;
    (*a)++;
    printf("%d\n", count = count + i);
}
```

Stampa:

3  
4  
5  
6

- 1) Manca definizione *main* ed *i*
- 3) Manca definizione *mywrite* e *count*
- 4) Manca definizione *count* in *main.c* perchè *count* ha linkage interno in *write.c* quindi non è visibile. Manca poi la dichiarazione con linkage esterno di *i* in *write.c*

Il punto 4) può essere corretto cambiando la tipologia del linkage di *count* (globale) in *write.c* da interno a esterno. Si fa eliminando la keyword “static”:

**int count = -3;**

Va inoltre inserita in *write.c* la dichiarazione di *i* con linkage esterno: **extern int i;**

In main.c:

- *mywrite* è dichiarata ed ha linkage esterno
- *count* è dichiarata ed ha linkage esterno
- *i* a riga 3 è un tentativo di definizione e ha linkage esterno
- *i* a riga 4 è ora definita e ha linkage esterno

In write.c:

- DOPO LA CORREZIONE: *count* a riga 3 è definita e ha linkage esterno
- *i* è dichiarata ed ha linkage esterno
- *mywrite* è definita e ha linkage esterno
- *a* locale in *mywrite* è definita e ha no linkage
- *count* locale in *mywrite* è definita e ha no linkage

5. **6 punti** Cerchiare le affermazioni vere dato:

*int a[7] = {21, -21, [3]=INT\_MAX, 65537, [6]=511}; short \*ptr = (short\*) a; char \*n = (char\*) a;* sapendo che i tre tipi usati occupano 4, 2 e 1 byte e  $65536 = 2^{16}$  (valori rappresentati in complemento a due e *little endian*). Rappresentare la zona di memoria in cui è memorizzato l'array.

- A.**  $n+5 \geq \&ptr[3]$ ; **(B)**  $*(n+5) > *(n+4)$ ; **C.**  $\&ptr[8] == ptr+9$ ; **D.**  $((int)(ptr+8)-(int)(\&a[2]) < 8)$ ;  
**(E)**  $*(ptr+1) == *(a+2)$

## SOLUZIONE ESERCIZIO 5

10101000	a[0]
00000000	
00000000	*(ptr+1)
00000000	
11010111	*(n+4)
11111111	*(n+5)
11111111	&ptr[3]
11111111	
00000000	*(a+2) e a[2]
00000000	
00000000	
00000000	
11111111	
11111111	
11111111	
11111110	
10000000	&ptr[8] o ptr+8
00000000	
10000000	ptr+9
00000000	
00000000	
00000000	
00000000	
11111111	
10000000	
00000000	
00000000	

## SOLUZIONE ESERCIZIO 3

```

1 Node* ritorna_dispari(Node* lista_input) {
2
3     if (lista_input == NULL) {
4         return NULL;
5     } else {
6         int counter = 0;
7         Node* pScan = lista_input;
8
9         Node* lista_output = NULL;
10        Node* lista_output_pLast = NULL;
11
12        while (pScan != NULL) {
13            if (counter % 2 == 0) {
14                Node* pNew = (Node*) malloc(sizeof(Node));
15                pNew->info = pScan->info;
16                pNew->pNext = NULL;
17
18                if (lista_output == NULL) {
19                    lista_output = pNew;
20                    lista_output_pLast = pNew;
21                } else {
22                    lista_output_pLast->pNext = pNew;
23                    lista_output_pLast = pNew;
24                }
25            }
26
27            pScan = pScan->pNext;
28            counter++;
29        }
30
31        return lista_output;
32    }
33 }
34

```

A = FALSO, (n+5) sta in una cella di memoria inferiore a (&ptr[3])

B = VERO, \*(n+5) == -1, \*(n+4) == -21 quindi -1 > -21

C = FALSO, ricordati che si sta guardando l'indirizzo di memoria e non il loro contenuto

D = FALSO, la differenza in byte tra i due puntatori è 8

E = VERO, \*(ptr+1) == 0, \*(a+2) == 0 quindi \*(ptr+1) == \*(a+2)