Prova scritta Programmazione I - 14 Gennaio 2020 - FILA A

Nome e Cognome: _____ Matricola: _____

1. [3 punti] Elencare le conversioni di tipo (... *da* ... a). Dato *USHRT_MAX = 65535*, scrivere il valore finale della variabile *b* sapendo che il carattere *a* ha valore 97 (in ASCII) e le altre lettere seguono in ordine alfabetico.

```
1  long int fun2(int p) {
2    return (p + 'g' - 'b'); }
3
4  int fun1(long p) {
5    char a= 'i';
6    return fun2(p - a + 'a'); }
7
8  int main(void) {
9    unsigned short a =  -1LL;
10   double b= fun1(a);
11 }
```

linea 9: -1LL da long long a unsigned short
linea 10: a da unsigned short a long int **b vale 65532.0**
linea 5: 'i' da int a char
linea 6: a da char a long, 'a' da int a long
linea 6: p-a-'a' da long a int
linea 2: p + 'g' - 'b' da int a long (ritorno funzione)
linea 6: risultato fun2 da long a int (ritorno funzione)
linea 10: risultato fun1 da int a double

linea 9: a è inizializzato a 65535 , –1 + (USHRT_MAX + 1)

2. [3 punti] Scrivere cosa stampa la seguente porzione di codice.

```
1  int i, s, c= 0, c1= 0, a= 4, k=0;
2  for(i=1; i<a; ++i, k=0) {
3    for(s=1; s<a-i; ++s) {
4      printf("-");
5      ++c; }
6
7    while(k != 2*i+1) {
8      if (c < a-1) {
9        printf("*-");
10       c+=1; continue;}
11     else {
12       ++c1;
13       printf("%d-", (k*c1-i));}
14     ++k; }
15   c1 = c = k = 0;
16   printf("\n"); }
```

--*--1-1-5-
-*-*--2-0-4-10-18-
***-*-*--3--1-3-9-17-27-39-**

3. [3 punti] Scrivere cosa stampa la seguente porzione di codice.

```
1  int a= 0xfe - 012;
2  while(a > 9 ? !!!0: !!(a+1, a-=2, --a)) {
3    printf("%d \n", a);
4    if (a + 2> 0x10) {
5      a= -3;
6      continue; }
7    a+= 4;
8  }
9  !(a+1) || a++;
10 printf("a: %d\n", a);
```

244
-6
-5
-4
-3
-2
-1
a: 1

4. [4 punti] Su foglio protocollo, scrivere la definizione di una funzione *crea_m* che prende come parametri *righe* e *colonne* (*int*), e crea una matrice righe×colonne riempiendo i valori da tastiera (*scanf*). Infine, la funzione ritorna la matrice. *Suggerimento: creare la matrice come un semplice array di righe×colonne di tipo* int*, ed utilizzare l'artimetica dei puntatori per indicizzare l'array come una matrice.*

ESERCIZIO 4 (main non richiesto)

```
int* crea (int row, int column) {
   int* m = (int*) malloc(row*column * sizeof(int));
   for (int i= 0; i < row; i++)
     for (int j= 0; j < column; j++)
       scanf("%d", m + (column *i) + j);

   return m;
}


int main()
{

   int*p = crea(2, 3);
   for (int i= 0; i < 2; i++)
     for (int j= 0; j < 3; j++)
        printf("%d", *(p + (3 *i) + j));
}
```

ESERCIZIO 5

```
void print_I(struct Node* pFirst) {
   if(pFirst == NULL)
     printf(”Lista vuota!”);
   else {
      counter = 1;
      struct Node* pScan = pFirst;
      while (pScan != NULL)
         if (counter % 2 != 1) {
            printf(”Info: %d\n", pScan->info);
            pScan = pScan -> pNext;
            counter++;
         }
         else {
            pScan = pScan -> pNext;
            counter++;
         }
   }
return;
}
```

5. 3 punti Data la seguente *struct* definire una funzione di nome *print_pari* che che prende come parametro una lista e stampa su video il valore del campo *info* per tutti gli elementi in posizione pari della lista: supposto il primo elemento in posizione 1, stampa invece quelli in posizione 2, 4, 6, etc (se presenti...).

```
1 struct Node {
2   int info= 0;
3   struct Node* pNext= NULL;
4 }
```

6. 3 punti Evidenziare con una freccia ciascun sequence point nel codice sottostante. Quanti effetti collaterali ci sono su a (supponendo $a = 1$) in totale? Scrivere una espressione con 3 effetti collaterali su una variabile a e 1 effetto collaterale su b, che NON generi un warning *multiple unsequenced modifications*, e una espressione che lo generi. Evidenziare i sequence point in entrambe le espressioni.

```
1   while(a++) {
2   a > 0? a+=1 : a-= 1;
3   a++ || a++;
4   a, a--, a+= 1;
5   }
6
```

La condizione del ciclo while non è mai falsa (uguale a 0), provocando così "infiniti" effetti collaterali
b= a++, a++, a+=1 (non genera warning)
b= a++ + a++, a+=1 (genera warning)
corretto anche 5, guardando il codice per una sola iterazione

7. 4 punti Dire quali compilazioni provocano errore a causa del linker (e perché): 1) *gcc –c write.c*, 2) *gcc –o main main.c*, 3) *gcc –o write write.c*, 4) *gcc –c main.c*, 5) *gcc write.c main.c -o main*. In caso il punto 5) ritorni un errore, descrivere come può essere corretto. Dopo aver corretto l'errore, che tipo di *linkage* hanno *count*, *i*, e *mywrite*, ed in quale file sono definite? Cosa stampa il programma?

main.c
```
int count= 4;
void mywrite(int count);

int main(void) {
  int count= 4;
  do {
     mywrite(count);
  }while(count >=0);
}
```

write.c
```
#include <stdio.h>
extern int count;

void mywrite(int a) {
   static int i= 1;
   printf("%d\n", count=
                  count - (a + i));
}
```

2) mywrite non definita, 3) funzione main e count non definiti

count ext in entrambi i file
mywrite ext in entrmbi i file
i ha no linkage

count definito in main.c
mywrite definito in write.c
i definito in write.c

Stampa
-1
-6
-11
-16
continua

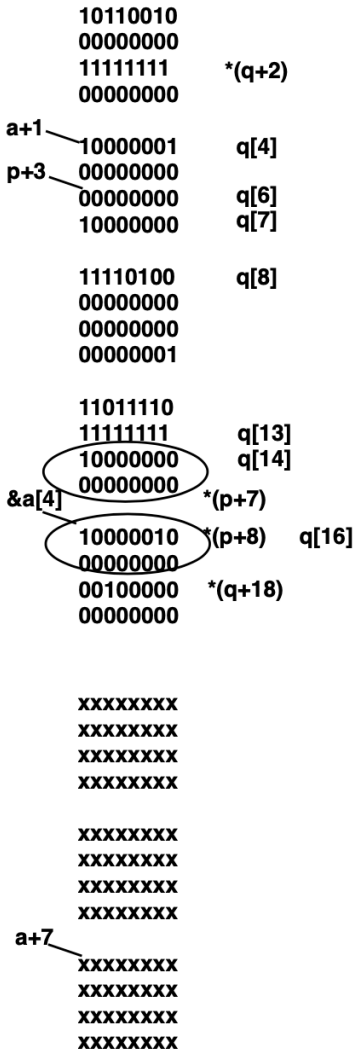
Nessun errore al punto 5) da correggere

8. 5 punti Cerchiare le affermazioni vere dato *int a[5]= {13+2*32, 129, [2]= INT_MIN + 47, 130939, 131072*2 +65}; short int *p = (short*) a; char *q= (char*) a; q[2]= -1; p[3]= 128*2;* sapendo che i tre tipi usati occupano 4, 2, e 1 byte, e $131072 = 2^{17}$ (valori rappresentati in *little endian* e complemento a due). Scrivere la mappa di memoria e giustificare le affermazioni (vere o false). Gli operatori | e & ritornano rispettivamente l'*or* e l'*and* bit-a-bit dei due operandi, $\sim$ è la negazione bit a bit, mentre >> (<<) rappresenta l'operatore di *shift* di n posizioni a destra (sinistra), inserendo 0 nelle posizioni eliminate (operazione fatta nel processore).

A. $(*(p+8) - *(q+2) - q[6]) > 65)$ B. $(\&a[4] - (a+1)) + *(q+18) - 7$ C. $(q[13] - q[14] + *(p+7))\%2$
D. $(*(q+2) \,|\, q[8]) + (q[7] \,\&\, q[13])$ E. $(((int)(a+7) - (int)(p+3)) + q[16])\%87$ F. $((q[8] >> 3) + \sim q[4])\%130$

9. 2 punti Su foglio protocollo, descrivere le differenze tra *dichiarazione* e *definizione* di variabile, con esempi.

ESERCIZIO 9



C : (-1 - (1) + 1) % 2 == -1 Quindi VERA

A : (65 - (-1) - 0) > 65 VERA

B : 3 + 4 - 7 == 0 FALSA
con &a[4] - (a+1) == 3 ci sono tre interi tra quei due puntatori

E : (22 + 65) % 87 == 0 FALSA
con (int) (a+7) - (int) (p+3) == 22 ci sono 22 byte tra quei due indirizzi

D : -1 + 1 == 0 FALSA
con
***(q+2) == 11111111 |**
q[8] == 11110100
dato che *(q+2) sono tutti 1, in or con qualsiasi altro operando il risultato è sempre 11111111 (che vale -1 in complemento a 2)
e
q[7] == 10000000 &
q[13] == 11111111
dato che q[13] sono tutti 1, in and con qualsiasi altro operando il risultato sempre l'altro operando, in questo caso 10000000 (che vale 1)

F: (5 + 126) % 130 == 1 VERA
con
q[8] vale 11110100, portato nei registri del processore viene rappresentato in big endian, cioè 00101111
00101111 >> 3 == 00000101
che letto in big endian vale 5
e
q[4] vale 10000001
~q[4] vale 01111110
che vale 126