Prova scritta Programmazione I FILA A - 25 Giugno 2018.

Nome e Cognome: _____ Matricola: _____

1. [3 punti] Riportare tutte le conversioni di tipo implicite. La variabile *e* riesce a rappresentare il valore di *d*? Spiegare perché.

E NON PUÒ RAPPRESENTARE CON PRECISIONE IL VALORE DI d
PERCHÉ d È IN PRECISIONE ESTESA E e È IN PRECISIONE SINGOLA

```
1 double f(float a) {return (a + 1);}
2
3 int main(void) {
4   short  a =  3; char b= 'a';
5   int c=  a + a + b;
6   long double d= f((b < c)? b : c);
7   float e= d;
8 }
```

LINEA 4 : 3 DA INT A SHORT
LINEA 5 : a(DUE VOLTE) E b, DA CHAR A INT (INT PROMOTION)
LINEA 6 : b DA CHAR A INT, c DA CHAR AINT
LINEA 1 : PARAMETRO PASSATO DA INT A FLOAT E 1 DA INT A FLOAT, VALORE RITORNATO DA FLOAT A DOUBLE
LINEA 6 : VALORE RITORNATO DA DOUBLE A LONG DOUBLE
LINEA 7 : d DA LONG DOUBLE A FLOAT

da short a int

2. [3 punti] Scrive una funzione *main()* che disegna su terminale la seguente figura composta di asterischi. Il numero di righe è preso come input da tastiera. Utilizare due cicli innestati.

```
*****
*   *
*   *
*   *
*****
```

https://tinyurl.com/yalvyqk6

3. [3 punti] Scrive una funzione *main()* che disegna su terminale la seguente figura composta di asterischi. Il numero di righe è preso come input da tastiera.

```
******
 ******
  ******
   ******
    ******
     ******
```

https://tinyurl.com/ybxlt8fv

4. [3 punti] Scrivere cosa stampa il seguente programma, sapendo che *a* si trova all'indirizzo 0x7ffee80188f8.

```
1 int a= 0xad, i= -11, *b= &a;
2 for (int* p= &i; (a-= 4, a++, (*p)++) ? (++(*p), (a--)-1) : ((*p)
      +=3, a-1); (*p)++) {
3   --a;
4    printf("%d %d \n", a, *p);
5    if (*p > 3) {
6      a= (!(-a) && a++) ? 6 : 4;
7      break; }
8    }
9    printf("%d %p %d\n", a, ((short*) b) + a, i);
```

168 -3
163 -6
158 -3
153 0
148 3
143 6

4 0x7ffee80188f8 6

5. [4 punti] Su foglio protocollo, scrivere una funzione *main()* e una funzione *double_array()*. La funzione *main()* legge da tastiera un array di n posizioni (con n letto da tastiera) ciascuna di tipo *int*. L'array viene passato a *double_array()* che crea in memoria dinamica un nuovo array di *int* di lunghezza $n \times 2$. Il nuovo array viene riempito dividendo aritmeticamente il primo elemento dell'array passato in due e memorizzando il risultato in posizione 0 e posizione 2 del nuovo array (esempio per $n == 2$): cioè, in prima posizione sia della prima metà, sia della seconda metà del nuovo array. Di seguito tutti gli altri elementi. Il nuovo array viene ritornato a *main()*, che lo stampa su schermo.

6. [3 punti] Su foglio protocollo, scrivere un solo file in cui siano presenti esattamente *i)* due tentativi di definizione di cui uno rimane dichiarazione e l'altro diventa definizione, *ii)* un identificatore con *internal linkage*, *iii)* tre identificatori con *external linkage* (di cui un identificatore di funzione), *iv)* due identificatori con *no linkage*, *v)* una variabile locale che risieda in memoria permanente, *vi)* una variabile locale che sia dichiarata.

7. [4 punti] Scrivere cosa stampa il seguente programma, sapendo che le variabili *a*, *b*, *c* in *main* sono memorizzate rispettivamente agli indirizzi $0x7ffeef2c970b$, $0x7ffeef2c980c$ e $0x7ffeef2c990e$.

```
1  static int a= 5;
2  int f(int* x, int y, int* z) {
3    int res= 2;
4    if ((a++,a-=2) > 1) {
5      a ? res=  (**&z+=3) + ++y, (y+= *x) : 2;
6      printf("%d %d  %p %d\n", *x, y, z+2, res);
7      res+= f(&y, *x+1, z);
8      return (-res);
9    }
10   else
11     return -res;
12 }
13
14 int main(void) {
15   int a= 3, b= 010, c= 0X4;
16   b= f(&b, a, &c);
17   printf("%d %d %d\n", a, b, c);
18 }
```

8 17 0x7ffeef2c9916 11
17 22 0x7ffeef2c9916 2C
22 36 0x7ffeef2c9916 27
3 -15 13

8. [3 punti] Data la seguente *struct Node*, definire una funzione ricorsiva di nome *rec_print* che per ogni elemento *Node* della lista stampi *i)* la somma dei campi *info* di tutti gli elementi *Node* precedenti in lista, e, *ii)* per ogni elemento, la somma dei campi *info2* di tutti i successivi elementi *Node* nella lista.

```
1  struct Node {
2    int info= 0;
3    int info2= 0;
4    struct Node* pNext= NULL;
5  }
6
```

9. [4 punti] Cerchiare le affermazioni vere dato *long long a[3]= {1536, -2, LLONG_MIN + 512};*
*short int *p = (short*) a; char *q= (char*) a; p[1]= 4097, p[3]= 4095-2, *(q+15)= 73, p[9]= 4096*4+1;*
sapendo che i tre tipi usati occupano 8, 2, e 1 byte, e $4096 = 2^{12}$ (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, mentre >> rappresenta l'operatore di *shift* di *n* posizioni a destra (ricordarsi che l'operazione è effettuata nei registri del processore...). Supporre che i tipi *char* siano tipi con segno: il bit più significativo uguale a 1 rappresenta un valore negativo.

A. $(q[8] \mid q[2]) + q[2] > 0$ B. $(*(p+5) - p[6]) \ \% \ 2$ C. $(((int)(p+12) - (int)(a+4)) + q[3]) \ \% \ 4$
D. $*((int*)(\&q[20])) + INT\_MAX > 0$ E. $(q[1] << 2) <= 16$ F. $(q[9] \mid q[10]) + q[2]$

TUTTE FALSE

ESERCIZIO 6

```
STATIC INT A = 3;
STATIC INT A;
INT B;
EXTERN INT C;
VOID FUN (INT D) {
    STATIC INT E = 2;
    EXTERN INT B;
}
```

ESERCIZIO 8

DA CHIAMARE LA PRIMA VOLTA CON S = = 0

```
INT REC_PRINT (STRUCT NODE *P, INT S) {
    IF (P -> PNEXT == NULL) {
        PRINTF("%d\N", S);
        PRINTF("%d\N", 0);
        RETURN (P -> INFO);
    }
    ELSE {
        PRINTF("%d\N", S);
        INT RES = REC_PRINT (P -> PNEXT, S + P -> INFO);
        PRINTF("%d\n", RES);
        RETURN (RES + P -> INFO);
    }
}
```

ESERCIZIO 5

```
INT* DOUBLE_ARRAY (INT* A, UNSIGNED L) {
UNSIGNED NEWL = L*2;
INT* A2 = (INT*) CALLOC (NEWL, SIZEOF(INT));
FOR (INT i=0; i<L; i++) {
        *(A2+i) = *(A+1) /2;
        *(A2+i+L) = *(A+1) /2;
}
RETURN (A2);
}
```

Esercizio 9

1 0 0 0 0 0 0 0
1 0 1 0 0 0 0 0
1 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0

0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
1 1 1 1 0 1 1 1
1 1 1 0 0 0 0 0

1 0 0 1 1 1 1 1
1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1

1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1
1 0 0 0 1 0 1 0

1 0 0 0 0 0 1 0
0 1 0 0 0 0 0 0
1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1

lq[20]

0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1

← *((int*)(lq[20]))

(D) *((int*)(lq[20])) È EQUIVALENTE A INT_MIN

INT_MIN + INT_MAX = -1

IN COMPLEMENTO A DUE E CON INTA 32 BIT

INT_MAX È $2^{31} - 1$

INT_MIN È -2^{31}

$-1 > 0$

FALSA

(E) q[1] NEL PROCESSO È RAPPRESENTATO COME

0 0 0 0 0 1 0 1

<< 2

0 0 0 1 0 1 0 0 == 40

$40 <= 16$

FALSA