

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.

```
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 }
```

VEDI PUA A

2. **3 punti** Scrivere una funzione *main()* che disegna su terminale il seguente triangolo di lettere.

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

VEDI PUA A

3. **3 punti** Scrivere una funzione *main()* che disegna su terminale il seguente triangolo di lettere. Il numero di righe è preso come input da tastiera.

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

VEDI PUA A

4. **3 punti** Scrivere cosa stampa il seguente programma, sapendo che *a* si trova all'indirizzo 0x7fee9289918.

```
1 int a= 0xac, i= -11, *b= &a;
2 for (int* p= &i; (a-= 3, 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	-9
164	-6
160	-3
156	0
152	3
148	6

4 0x7fee9289918 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* × 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.

VEDI PUA A

VEDI PUA A

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 *0x7ffeb7d296a*, *0x7ffeb7d297a* e *0x7ffeb7d297b*.

```
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 12 0x7ffeb7d2983 11  
 12 22 0x7ffeb7d2983 20  
 22 36 0x7ffeb7d2983 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
```

VEDI PUA A

9. **4 punti** Cerciare le affermazioni vere dato  $\text{long long } a[3] = \{1536, -2, \text{LLONG\_MIN} + 512\};$   
 $\text{short int } *p = (\text{short} *) a; \text{ char } *q = (\text{char} *) a; p[1] = 4097, p[3] = 4095-2, *(q+15) = 73, p[9] = 4096*4+1;$   
 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.  $((p+5) - p[6]) \% 2$  B.  $(q[8] | q[2]) + q[2] > 0$  C.  $(q[9] | q[10]) + q[2]$  D.  $((\text{int})(p+12) - (\text{int})(a+4)) + q[3] \% 4$  E.  $(q[1] << 2) <= 16$  F.  $*((\text{int}*)&q[20]) + \text{INT\_MAX} > 0$

TUTTE FALSE VEDI PUA A