



Nome e Cognome: _____

Matricola: _____

1. **3 punti** Scrivere una espressione con 3 effetti collaterali su una variabile a e 2 effetti collaterali su b , che *NON* generi un warning *multiple unsequenced modifications*, evidenziando i *sequence point* con una freccia. Scrivere poi una seconda espressione con 2 effetti collaterali su a e due effetti collaterali su b che invece generi tale warning solo a causa di a (ma non a causa di b), evidenziando anche in questo caso i *sequence point*.

1) $a=3, a++, a++, b--, b=3$

 2) $a++ + a++, b--, b=3$


2. **3 punti** Cerchiare i seguenti punti solo se veri, dato $\text{int } a=2, *const p = \&a$; e motivare le risposte su foglio protocollo. **(A.)** $\&a$ è un *rvalue*; **(B.)** $**\&p$ è un *lvalue*; **(C.)** $a = 2, a++$; non genera warning; **(D.)** p ha tipo *int* (costante); **(E.)** $!(*p) == -2$; **(F.)** $p = NULL$ genera errore.
3. **6 punti** Riportare nel riquadro a fianco cosa stampa la seguente porzione di codice, sapendo che l'indirizzo di a è `0x7ff7beca94f8`.

```
1 int a= 0x1e, b = 06, *c= &a;
2 do {
3     for (int i= 2; i ? —i : !!0 ; i—) {
4         a/=2; b = a;
5         printf("%d %d\n", a, i);
6         if (a == b) break;
7     }
8 } while (a—, a && b);
9 printf("%d %p %p\n", a, c, ((short*) c) - 1);
```

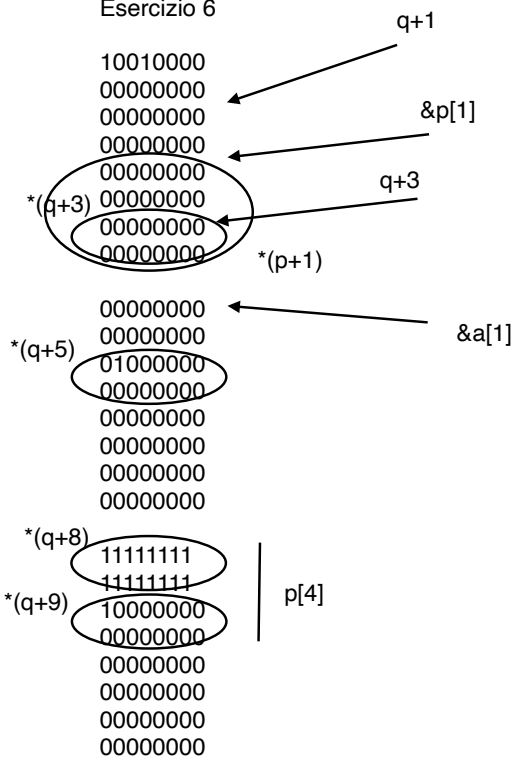
15 1
 7 1
 3 1
 1 1
 0 0x7ff7beca94f8 0x7ff7beca94f6

4. **6 punti** Data la seguente *struct* scrivere la definizione una funzione di nome *inverti_lista* che prende come parametro una lista e ritorna un'altra lista (creata nella funzione) che contiene gli stessi valori (campo *info*) della lista passata, ma in ordine inverso: se la lista passata è 7-3-1 la lista ritornata conterrà 1-3-7.

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

5. **5 punti** Su foglio protocollo, scrivere due file di nome *main.c* (che contiene la definizione di *main*) e *write.c*: *main.c* non può esser compilato (ed eseguito) da solo per produrre un eseguibile, ha bisogno di essere collegato a *write.c*. In *main.c* ci deve essere un tentativo di definizione che rimane dichiarazione e una variabile locale alla funzione *main* con linkage esterno. Il file *write.c* deve contenere una definizione con linkage esterno, una con linkage interno, e una *no linkage*. Evidenziare i punti precedenti nei due file. Scrivere inoltre i due comandi *gcc* per creare i due file oggetto, che devono compilare senza errore.
6. **7 punti** Cerchiare le affermazioni vere dato $\text{long long } a[3] = \{9, 131072, 131071\}$; $\text{int } *p = (\text{int}*) a$; $\text{short int } *q = (\text{short}*) a$; sapendo che i tre tipi usati occupano 8, 4, e 2 byte, e $131072 = 2^{17}$ (valori rappresentati in *little endian* e complemento a due). Rappresentare la mappa di memoria e giustificare per ogni risposta perché vera o falsa. **(A.)** $q+3 > \&p[1]$ **(B.)** $*(p+1) == *(q+3)$; **(C.)** $*(q+5) - *(q+9) == 1$; **(D.)** $*(q+8) > 0$; **(E.)** $p[4] == 131071$. **(F.)** $((\text{int})(\&a[1]) - (\text{int})(q+1)) < 6$.

Esercizio 6



Esercizio 4

```
struct Node* inverti_lista(struct Node* lista) {
    struct Node* nuova_lista= NULL;
    if (lista != NULL) {
        struct Node* pScan = lista;
        do {
            struct Node* nuovo_nodo = (struct Node*) malloc(struct Node);
            nuovo_nodo -> info = pScan -> info;
            if (nuova_lista == NULL) {
                nuovo_nodo -> pNext = NULL;
                nuova_lista = nuovo_nodo;
            }
            else {
                nuovo_nodo -> pNext = nuova_lista;
                nuova_lista = nuovo_nodo;
            }
            pScan = pScan -> pNext;
        }while(pScan!= NULL);
    }
    return nuova_lista;
}
```

Esercizio 3

- A: &a è un indirizzo di memoria, quindi un valore intero, che non può stare a sinistra di un uguale
- B: equivale a *p, cioè la variabile a che può stare quindi a sinistra di un uguale
- C: le modifiche su a sono sperate da un sequence point (,)
- D: p ha tipo puntatore ad intero (costante)
- E: 0 == -1 è falso
- F: genera errore perché p è un puntatore costante (a cui non si può cambiare valore)

- A: Vero, se a è 1000, q+3 vale 1006 e &p[1] vale 1004
- B: Vero 0 == 0
- C: Vero 2-1 == 1
- D: Falso *(q+8) == -1
- E: Vero è il valore dei primi 4 byte del terzo elemento dell'array
- F: Falso, la differenza fa esattamente 6

Esercizio 5

```
-main.c-
int ret1(void);
int a= 0;
int a; //tent def che rimane dichiarazione

int main(void) {
    extern int a; //var locale link esterno
    a = ret1();
}

-write.c-
static int b= 0; //def link interno

int ret1() { //def link esterno
    int c = 1;
    return c; //def no linkage
}
```