Prova scritta Programmazione I - 28 Gennaio 2019 - FILA B

Nome e Cognome: _____ Matricola: _____

1. [3 punti] Riportare le conversioni di tipo implicite, scrivere il valore di finale di c e d, dato $USHRT\_MAX = 65535$ e $UINT\_MAX = 4294967295$. In caso non sia possibile stabilire il valore di d, spiegare perché.

```
1 double f(float a) {return (a - 2);}
2
3 int main(void) {
4   unsigned short a =  -65533;
5   unsigned int b=  -1;
6   float c= f((b < a)? b : a);
7   double d= UINT_MAX - 4;
8 }
9
```

c vale 1.0
d vale 4294967291
rappresentabile con un double che ha precisione doppia, rappresenta 15 cifre
a convertito a uint vale (65535 + 1) - 65533 == 3
b convertito a uint (4294967295 + 1) -1 == 4294967295
linea 4: -65533 da int a unsigned int
linea 5: -1 convertito da int a unsigned int
linea 6: in b < a, a da unsigned short a unsigned int (risultato comparazione convertito a int)
linea 6: la condizione è falsa, a convertito da ushort a uint. (l'espressione condizionale ritorna un solo tipo)
linea 6: a convertito poi da uint a float
linea 1: -2 convertito da int a float
linea 1: il risultato convertito da float a double
linea 6: f(a) convertito da double a float
linea 7: UINT_MAX - 4 convertito da long oppure long long a double

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

```
1 int n= 5, k = 1, b, i, j;
2 for(i=0; i<n; i++) {
3 for(b=0; b <= n-i; b++)
4   printf("-");
5   for(j=0; j < i; j++) {
6   if (j==0 || i==0) k = 1;
7   else
8     k*= (i-j+2)/j-1;
9 printf("-%d", k); }
10 printf("\n"); }
```

------1
-----1-2
----1-3-0
---1-4-4-0

3. [3 punti] Scrivere cosa stampa la seguente porzione di codice. Che valore contiene la variabile b alla fine del programma?

```
1 int a= 0x77, i= -5, *b= &a;
2 for (int* p= &i; (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++) ? 2 : 5;
7     break; }
8   else continue; }
9 printf("%d\n", a);
```

118 -3
117 0
116 3
115 6
5

4. [3 punti] Su foglio protocollo, scrivere la definizione di una funzione *arrayDiff* che prende due array di *int* come parametri, controlla che abbiano la stessa lunghezza, e ritorna la somma delle differenze tra elementi in posizioni corrispondenti: $a[0] - b[0] + a[1] - b[1] + \ldots$. Trattare i casi in cui può accadere un errore.

5. [4 punti] Data la seguente struttura, definire una funzione di nome *ins_head* che prende come parametri un puntatore a *struct Node* (il puntatore al primo elemento della lista) e un valore *int*, e inserisce un nuovo elemento in testa alla lista (cioè come primo elemento), con il campo *info* assegnato al valore passato. La funzione non crea il nuovo elemento ed esce stampando un errore se il campo *info* del nuovo elemento è minore del campo *info* dell'elemento successivo in lista.

VEDERE CORREZIONE FILA A PER ESERCIZI 4,5,6,7

```
1  struct Node {
2  int info;
3  struct Node* pNext;
4  }
5
```

6. [3 punti] Su foglio protocollo, scrivere una funzione ricorsiva di nome *findMax* che ritorna la massima cifra del valore di tipo *unsigned int* passato come parametro. Per esempio, chiamata con 395 deve ritornare 9, con 54321 deve ritornare 5.

7. [3 punti] Su foglio protocollo, scrivere due file, *main.c* (che contiene la funzione *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 esattamente una definizione con *linkage* esterno e una con *linkage* interno; questo file deve inoltre contenere esattamente una definizione con *no linkage*. Scrivere inoltre i due comandi per creare i file oggetto dei due file.

8. [4 punti] Scrivere cosa stampa il seguente programma.

```
1 int f1(int* x, int y, int* z) {
2   static int a= -1;
3   a++;
4   int res= -2;
5   if (a < 3) {
6     res= ++y + ++(*x), (y+= *z+1);
7     res+= f1(x, y, z);
8     printf("%d %d %d %d\n", *x, y, *z, res);
9     return (res); }
10  else
11    return res; }
12
13 int main(void) {
14   int a= 3, b= 4, c=-3;
15   b= f1(&a, b, &c);
16 p  rintf("%d %d %d\n", a, b, c);
17 }
18
```

6 1 -3 7
6 2 -3 16
6 3 -3 25
6 25 -3

9. [4 punti] Cerchiare le affermazioni vere dato *int a[4]= {7+4*64, INT_MIN + 39, [2]= 131002, 131072/2+111}; short int *p = (short*) a; char *q= (char*) a; *(q+3)= -1; *((short int*) &q[5])= 257;* 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, inserendo 0 nelle posizioni eliminate (operazione fatta nel processore).

(A.) $p[3] > SHRT\_MIN$ (B.) $((\&a[4]-a)+-p[5])\ \%2$ C. $q[5]-q[6]+q[10]-q[14]$ (D.) $(*(q+3)\ \&\ q[1])+\ q[2]$
(E.) $((q[12] >> 4)\ |\ q[4]) >= 35$ (F.) $(((int)\ (a+2) - (int)\ \&q[2]) + *(q+14))\ \%2)$

Vedere FILA A per mappa di memoria e correzione dei vari punti