Prova scritta Programmazione I FILA B - 23 Febbraio 2018.

Nome e Cognome: _____ Matricola: _____

1. [3 punti] Riportare le conversioni di tipo implicite, dare il valore di finale di d dato $INT\_MAX = 2147483647$. In caso non sia possibile stabilire il valore di d, spiegare perché. Scrivere la gerarchia dei tipi interi ed in virgola mobile (*tipo1* ">" *tipo2* significa che *tipo1* è più alto in gerarchia).

Linea 7: INT_MAX - 2 da int a float

UN TIPO FLOAT HA PRECISIONE SINGOLA: PUÒ SICURAMENTE RAPPRESENTARE VALORI ON 6 CIFRE MENTRE INT_MAX NE HA 10. IL VALORE DI D DIPENDE DALL'APPROSSIMAZIONE; NON SO DIRLO

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

LINEA 4: 4 DA INT A UNSIGNED LONG INT
LINEA 6: b<a -> b DA INT A UNSIGNED LONG DIVENTA QUINDI UN VALORE MOLTO ALTO (DIPENDE SE LONG È 4 O 8 BYTE). NO VIENE PASSATO A LINEA 6: A DA UNSIGNED LONG A FLOAT
LINEA 1: 1 DA INT A FLOAT
LINEA 1: a-1 DA FLOAT A DOUBLE
LINEA 6: RISULTATO F DA DOUBLE A FLOAT

SLIDE 7 CONVERSIONI DI TIPO: _Bool < char < SIGNED < INT < LONG < LONG LONG < FLOAT < DOUBLE < LONG DOUBLE

2. [3 punti] Scrivere cosa stampa il seguente programma.

```
1  int r= 6, c = 1, s, i, j;
2  for(i=0; i<r; i++) {
3    for(s=1; s <= r-i; s++)
4      printf("-");
5
6    for(j=0; j <= i; j++) {
7      if (j==0 || i==0)
8        c = 3;
9      else
10       c = c*(i-j+2)/j;
11     printf("-%d", c);
12   }
13 printf("\n");
14 }
```

---3
---3-6
---3-9-9
---3-12-18-12
---3-15-30-30-15
---3-18-45-60-45-18

3. [3 punti] Scrivere cosa stampa il seguente programma, sapendo che a si trova all'indirizzo 0x7ffee4399fff.

```
1  int a= 0x1b, i= 03, *b= &a;
2
3  for (int* p= &i; (a-= 1) ? (*p++, --a) : ((*p)+=2, a); *p++) {
4    a= (a - i);
5    printf("%d %d OK\n", a, *p);
6    if (a <= 0) {
7      a= 1;
8      continue; }
9  }
10 printf("%d %p\n", a, ((short*) b) + 1);
```

21 4 OK
13 6 OK
3 8 OK
-9 10 OK
1 0x7FFEE4399A001

SE IL PUNTATORE È CONVERTITO A PUNTATORE A SHORT SE AGGIUNGO 1 AGGIUNGO 2 BYTE

4. [4 punti] Su foglio protocollo, scrivere la definizione di una funzione *matrix_sum_diagonals* che somma gli elementi sulle due diagonali di una matrice $n \times n$ (matrice di *int*) e ritorna un valore intero 1 se la somma è uguale, o 0 altrimenti. Se la matrice è *identica* (tutti gli elementi della diagonale principale sono costituiti dal numero 1, mentre i restanti elementi sono 0), la funzione stampa a schermo un messaggio appropriato.

SU FOGLIO SEGUENTE

5. [3 punti] Su foglio protocollo, descrivere cosa sono i *lvalue* e i *rvalue*, compresi i *lvalue* non modificabili. Fornire esempi di espressioni che li contengono.

SLIDE 7-12 ESPRESSIONI E OPERATORI

6. [3 punti] Data la seguente struttura, definire una funzione *ricorsiva* di nome *recursive_list* che crea in memoria dinamica una lista di strutture di lunghezza l, valore di tipo *unsigned int* preso come parametro.

ESERCIZIO 8

```
DOUBLE c = 0.0;
EXTERN VOID MY_FUNC(INT, INT*);

DOUBLE AREA (DOUBLE A) {
    RETURN (A * A);
}

INT MAIN(VOID) {
    INT A;
    MY_FUNC(A, &A);
}
```

ESERCIZIO 4

```
UNSIGNED INT MATRIX_SUM_DIAGONALS (INT N, INT M[N][N]) {
    INT SUM1=0, SUM2=0;
    FOR (INT i = 0; i < N; i++) {
        SUM1 += M[i][i];
        SUM2 += M[i][N-i-1];
    }
    UNSIGNED INT RES = (SUM1 == SUM2);
    UNSIGNED INT IDENTICAL = 1;
    FOR (INT i=0; i < N; i++)
        FOR (INT J = 0; J < N; J++)
            IF (i == J) {
                IF (M[i][J] != 1)
                    IDENTICAL = 0;
            }
            ELSE
                IF (M[i][J] != 0)
                    IDENTICAL = 0;
    IF (IDENTICAL)
        PRINTF ("MATRICE IDENTICA");
    RETURN RES;
}
```

```
struct Node {
  int info;
  struct Node* pNext;
};
```

```
STRUCT NODE* RECURSIVE_LIST (INT L) {
IF (L == 0)
    RETURN (NULL);
IF (L >= 1) {
    STRUCT NODE *P = (STRUCT NODE*) MALLOC (
                        SIZEOF (STRUCT NODE));
    P -> PNEXT = RECURSIVE_LIST (L-1);
    RETURN (P); } }
```

7. [4 punti] Scrivere cosa stampa il seguente programma, sapendo che le variabili *a*, *b*, *c* in *main* sono memorizzate rispettivamente agli indirizzi $0x7ffeeb7d2964$, $0x7ffeeb7d2970$ e $0x7ffeeb7d2974$.

```
1  static int a= 3;
2  int f1(int* x, int y, int* z) {
3    int res= 1;
4    if (a-- > 0) { \\ a meno meno
5      a ? res= ++y + ++(*z), (y+= *x) : 2;
6      printf("%d %d %p %d\n", *x, y, z, res);
7      res+= f1(&y, *x, z);
8      return (res);
9    }
10 else
11   return res; }
12
13 int main(void) {
14   int a= 3, b= 4, c= 4;
15   b= f1(&b, a, &c);
16   printf("%d %d %d\n", a, b, c);
17 }
```

```
4  8  0x...2974  9
8  13 0x...2974  11
13 8  0x...2974  1
3 22 6
```

8. [4 punti] Per ogni identificatore di variabile e funzione ad ogni linea, scrivere se corrisponde ad una definizione o dichiarazione, ed il tipo linkage associato. Scrivere un file *file1.c* compilabile senza errori con il file sottostante (di nome *file2.c*) attraverso il comando *gcc -o eseguibile file1.c file2.c*. L.I = LINKAGE INTERNO N.L. = NO LINKAGE L.E. = " ESTERNO

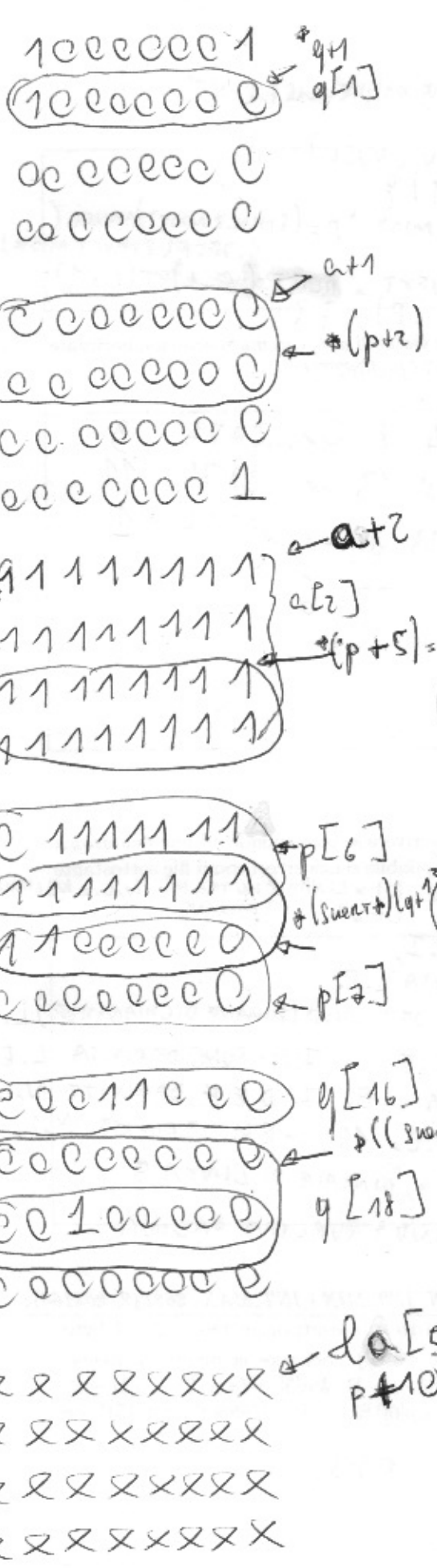
```
1  static double a;
2  extern double area(double);
3  typedef int pippo;
4  int b;
5  extern pippo b= 1;
6  extern double c;
7
8  extern int* my_func(int d, int* e) {
9    static float f;
10   double g, h[d];
11   extern pippo b;
12   g= area(c);
13   // Comandi...
14 }
```

1: A DEFINITA L.I.
2: AREA DICHIARATA L.E.
4: B TENTATIVO DI DEFINIZIONE (RIMANE DICHIARAZIONE) L.E.
5: B DEFINITA L.E. 7: MY_FUNC DEFINITA L.E.
6: C DICHIARATA L.E. 7: D E E DEFINITE N.L.
9: F DEFINITA N.L. 10: G E H DEFINITE N.L.
11: B DICHIARATA, LINKAGE DI LINEA 5

CONTINUA SULLA PAGINA

9. [4 punti] Cerchiare le affermazioni vere dato *int a[5]= {129, INT_MIN, INT_MIN | INT_MAX, 262142, 262168}; q[1]= 1; short int *p = (short*) a; char *q= (char*) a;* sapendo che i tre tipi usati occupano 4, 2, e 1 byte, e $262144 = 2^{18}$ (valori rappresentati in *little endian* e complemento a due). Scrivere la mappa di memoria e giustificare le affermazioni (vere o false). A. $a[2] + *(q+1)$ B. $\&a[5] - (a+2) - q[1] - 2$ C. $(((int)(p+10) - (int)(a+1)) + q[16] - q[18]) \,\%\, 6$ D. $(*(p+5) - p[6]) \,\%\, 1$ E. $*((short*)(q+13)) == *((short*)(q+17))$ F. $*(p+2)\,?\,p[7] : !(p[4]+p[5]+p[7])$

TUTTI FALSI



(A) a[2] + *(q+1) == 0 FALSA
-1 + 1

(B) &a[5] - (a+2) - q[1] - 2 == 0
3 - 1 FALSA

(C) (INT)(p+10) - (INT)(a+1) + q[16] - q[18]
16 + 24 - 4
== 36 % 6 = 0 FALSA

(D) *(p+5) - p[6] % 1 == 0
-1 - (-2) FALSA

(E) *((*sconst*)(q+13)) == *((*sconst*)(q+17))
1023 == 1024 FALSA

(F)
*(p+2) == 0
QUINDI
!(p[4] + p[5] + p[7])
!(-1 + (-1) + 3) = = !1 == 0
FALSA