PUC-Rio – Software Básico – INF1018
Prova 1 – 16/04/2011

Atenção: Não esqueça de colocar o seu nome e número de matrícula na(s) folha(s) de respostas. Seja claro e objetivo. **Boa sorte!**

1. (2,5 pontos) Considere o programa C a seguir (as reticências na inicialização de **x** são propositais):

```
#include <stdio.h>
void dump(void *p, int n) {
  unsigned char *p1 = (unsigned char *)p;
  while (n--) {
    printf("%p - %02x\n", p1, *p1);
    p1++;
  }
}
struct X {
  int i;
  char c;
  short s;
} x[2] = {{...}, {...}};

int main(void) {
  dump(x, 24);
  return 0;
}
```

Imagine que, ao ser executado em uma máquina de arquitetura IA-32 (*little-endian*), obedecendo as convenções de alinhamento do Linux, esse programa imprimiu a saída a seguir:

```
0x804a014 - b0
0x804a015 - 04
0x804a016 - 00
0x804a017 - 00
0x804a018 - 9c
0x804a019 - 00
0x804a01a - ff
0x804a01b - 03
0x804a01c - 00
0x804a01d - 80
0x804a01e - ff
0x804a01f - ff
0x804a020 - 3a
0x804a021 - 00
0x804a022 - 2c
0x804a023 - 01
0x804a024 - 00
0x804a025 - 00
0x804a026 - ff
0x804a027 - 00
0x804a028 - 02
0x804a029 - 00
0x804a02a - 10
0x804a02b - 00
```

Analisando a saída do programa e a declaração de `x` (um *array* de 2 estruturas), substitua as "reticências" na inicialização de `x` com os valores dos campos respectivos. Expresse esses valores

em notação decimal. **Explique** como você chegou a esses valores (mostre suas contas e a posição dos valores na saída do dump). Valores sem explicação não serão considerados!

ATENÇÃO: O programa pede que o dump imprima 24 bytes. Esse número é suficiente para mostrar todo o conteúdo do *array*, mas **podem ser exibidos bytes a mais!**

2. (2,5 pontos) Suponha as declarações abaixo, usadas no primeiro trabalho:

```
#define NUM_BYTES 16
typedef unsigned char BigInt[NUM_BYTES};
```

Ou seja, uma variável do tipo BigInt é representada por um array, que deve ser interpretado como um único inteiro de 128 bits, **em complemento a dois**, seguindo a ordem little-endian.

Implemente em C uma função que retorna o maior dentre dois BigInt. Essa função deverá ter a assinatura abaixo:

```
void bi_max(BigInt res, BigInt a, BigInt b);
```

Se for conveniente, você poderá utilizar funções auxiliares. Nesse caso, você deverá mostrar também esse código.

3. (2,5 pontos) Traduza a função `boo` abaixo para assembly IA-32 do gcc/Linux (visto em sala), utilizando as regras usuais de alinhamento, passagem de parâmetros e retorno de resultado em C. Comente o seu código!

(Não se preocupe se você não entender o que as função faz, apenas traduza-a literalmente).

```
struct L {
  int f;
  int v;
  struct L *next;
};

int foo(int i, int f);

void boo(struct L *l, int i) {
  while (l != NULL) {
    if (l->v == 0)
      l->v = foo(i, l->f);
    l = l->next;
  }
}
```

4. (2,5 pontos) Traduza a função `min` em assembly IA-32 abaixo para uma função equivalente em C.

```
int min(int a[], int n) {
  int m = INT_MAX;  /* maior valor de um inteiro com 32 bits */
  int i;
  for (i = 0; i < n; i++) {
    if (a[i] < m)
      m = a [i];
  }
  return m;
}
```