PUC-Rio – Software Básico – INF1018
Prova 1 – 02/10/2012 – Turma 3WB

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) No nosso primeiro trabalho, implementamos um tipo "inteiro grande" representado por um **array de bytes** que é interpretado como um único valor inteiro de N bits, em **complemento a dois**, seguindo a ordenação **little-endian**.

   Supondo uma implementação de inteiros grandes onde o número de bits N é pré-determinado e igual a **128**, escreva em C uma função que compare dois desses valores, com o seguinte protótipo:

   ```
   typedef unsigned char *BigInt;
   int bi_compare(BigInt a, BigInt b);
   ```

   Essa função deverá retornar:

   - -1 se o "inteiro grande" a é **menor** que o "inteiro grande" b
   - 0 se o "inteiro grande" a é **igual** ao "inteiro grande" b
   - 1 se o "inteiro grande" a é **maior** que o "inteiro grande" b

2. (2,5 pontos) Considere o programa C a seguir:

   ```
   #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 c1;
     struct X *p;
     short s;
     char c2;
   };
   int main(void) {
     struct X x[2];
     x[1].i = -2049 << 4;
     x[1].c1 = 'f' + 3;
     x[1].p = &(x[0]);
     x[1].s = 514 | 0x55aa;
     x[1].c2 = -125;
     dump((void *)&(x[1]), sizeof(struct X));
     return 0;
   }
   ```

   Considerando que o array x foi alocado no endereço de memória 0xff948234 diga o que o programa irá imprimir quando executado, explicando como você chegou a esses valores (valores sem contas e explicações NÃO SERÃO CONSIDERADOS!). Se houver posições de *padding*, indique seu conteúdo com **PP**.

   Considere que o valor do caracter 'a' na tabela ASCII é 97 (decimal), e suponha que a máquina de execução é *little-endian*, e que as convenções de alinhamento são as do Linux no IA-32.

3. Traduza as funções `f` e `g` abaixo para assembly IA-32 do gcc/Linux (visto em sala), utilizando as regras básicas de alinhamento, passagem de parâmetros, retorno de resultado e uso de registradores em C. **Comente o seu código!**

   (Não se preocupe se você não entender o que as funções fazem, apenas traduza-as literalmente).

   (a) (2,5 pontos)

```
void f(int a[], int b[], int c[], int n) {
 for (i = 0; i < n; i++) {
   if (a[i] > b[i])
     c[i] = a[i];
   else
     c[i] = b[i];
 }
}
```

   (b) (2,5 pontos)

```
#define NULL 0

struct No {
  int key;
  int val;
  struct No *left;
  struct No *right;
};

struct No *g(struct No *n, int k) {
  if (n == NULL)
    return NULL;
  else {
    if (n->key == k)
      return n;
    if (k < n->key)
      return g(n->left);
    else
      return g(n->right);
  }
}
```