PUC-Rio – Software Básico – INF1018
Prova Final – 01/12/2011

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:

```
#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 {
  char c;
  int i;
  short s;
} x = {-2, 1025, (260 >> 2)};

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

   Considerando que x seja armazenado no endereço de memória 0x2A000008, mostre o que o programa irá imprimir quando executado, **explicando como você chegou a esses valores**. Valores sem contas e/ou explicações não serão considerados!
   Suponha que a máquina de execução é *little-endian*, e que as convenções de alinhamento são as do Linux no IA-32. Escreva 'LL' nos bytes usados para *padding*.

2. Considere a representação IEEE de ponto flutuante vista no curso.

   (a) (1,5 pontos) Escreva uma função C que recebe um *double* e, usando instruções de manipulação de bits, retorne um inteiro que representa o *valor real* do expoente desse *double*. O protótipo dessa função é:

```
int expoente (double d);
```

   Para acessar a representação binária do parâmetro *double* você pode usar uma `union` como a mostrada a seguir:

```
typedef union {
  double d;
  unsigned int i[2];
} U;
```

   (b) (1,0 pontos) Usando a função anterior, e sem utilizar comparações de ponto flutuante, escreva uma função C que recebe um *double* e retorna 1 se o valor absoluto do número for menor que 1 e 0 caso contrário. O protótipo dessa função é:

```
int abs_menor_que_um (double d);
```

3. Traduza as função `foo` e `boo` a seguir para assembly IA-32, utilizando as regras usuais de alinhamento, passagem de parâmetros, retorno e salvamento de registradores para C/Linux.
Comente o seu código!

   (a) (2,5 pontos)

```
int foo(int a[], int n) {
  if (n == 0)
    return 0;
  else
    return foo(a, n-1) + a[n-1];
}
```

   (b) (2,5 pontos)

```
double g(float x);

double boo (float *pf, int n) {
  double s = 0.0;
  while (n--) {
    s += g(*pf);
    pf++;
  }
  return s;
}
```