PUC-Rio – Software Básico – INF1612 — Prova 2 – 30/12/06

1. Traduza as funções `soma` e `zee` abaixo para assembly IA-32 (o assembly visto em sala), utilizando as regras usuais de alinhamento, passagem de parâmetros e retorno em C/Linux. Comente seu código. (Não se preocupe em saber o que as funções fazem, apenas traduza-as literalmente!)

   (a) (2,5 pontos)

   ```
   struct SNo {
     double val;
     struct SNo *prox;
   };
   double soma (struct SNo *v) {
     if (v == 0) return 0;
     else return v->val + soma(v->prox);
   }
   ```

   (b) (2,5 pontos)

   ```
   char foo (char c);
   double zee (double d) {
     return foo(d);
   }
   ```

2. (3,0 pontos) Considere o trecho de código esquematizado abaixo:

   ```
   int foo (double d, float f);
   int bar (...) {
     foo (5.5, -123);
     a++;
     ...
   }
   ```

   Suponha que no momento em que o controle chegar ao label `foo` (ou seja, logo após executar a instrução `call foo`) o valor de `%esp` seja `0x00237A20`, e que o endereço da instrução seguinte à chamada a `foo` (referente ao código de `a++`) seja `0x0010B3D4`. Mostre o conteúdo da memória nesse momento, entre os endereços `0x00237A20` e `0x00237A2F`. O conteúdo deve ser mostrado byte a byte, em hexadecimal.

   Mostre suas contas e o raciocínio usado para chegar à sua resposta!

3. (2,0 pontos) Liste todos os símbolos exportados (definidos pelo arquivo) e importados (esperados de outros arquivos) pelo módulo abaixo, ou seja, o que aparecerá como `D` (dados) ou `T` (texto, ou código), na primeira categoria, e, na segunda categoria, o que aparecerá como `U` (undefined), na listagem do `nm`.

   ```
   #include <math.h>

   double a = 3.0;
   static int c = -1;
   extern int  b;

   int magica (char *x);

   static void foo (int p) {
     float f = 0.0;
     a = 1.0 + p + sin(f);
   }

   int bar (char *p1) {
     return magica (p1) + c;
   }
   ```