PUC-Rio – Software Básico – INF1018
Prova 1 – Turma 3wa – 2/10/2015

1. (2,5 pontos) Considere o programa C a seguir:

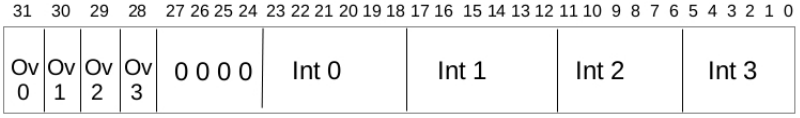
```
#include <stdio.h>
void dump (void *p, int n) {
  unsigned char *p1 = (unsigned char *) p;
  while (n--) {
    printf("%p - %02x\n", p1, *p1);
    p1++;
  }
}

struct X {
  char c1;
  char *c2;
  int num;
  short a;
};

int main (void) {
  char a[] = "abc";
  struct X x1 = {'b', a, 38<<8,-2050};
  dump (&a, sizeof(a));
  dump (&x1, sizeof(struct X));
  return 0;
}
```

   Supondo que `x1` seja alocado na posição de memória 0xbf84b290 e a no endereço de memória 0xbf84b2a0, diga o que esse programa irá imprimir quando executado. Coloque **PP** nas posições correspondentes a *padding*.

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

   (ATENÇÃO: mostre como você chegou aos valores exibidos. Valores sem contas **NÃO** valem ponto!).

2. (2,0 pontos) Considere o vetor de pequenos inteiros do nosso primeiro trabalho. Representamos esse vetor utilizando um inteiro sem sinal com o seguinte formato:



   - os bits 31, 30, 29 e 28 indicam a ocorrência de *overflow* na geração dos valores armazenados nos campos Int0 a Int3.
   - os bits 27 a 24 contém o valor 0 (zero)
   - os bits 23-18 (Int0), 17-12 (Int1), 11-6 (Int2) e 5-0 (Int3) armazenam os pequenos inteiros (valores inteiros de 6 bits, em complemento a 2).

Implemente em C uma função `vs_insere`, que substitui um dos números armazenados no vetor por um novo valor. A função deve setar o campo indicado com o valor inteiro truncado apropriadamente, e colocar também o valor correto no bit de overflow (indicando se há perda de valor no truncamento do novo item). A função pedida tem o protótipo:

```
typedef unsigned VetSmallInt;
int vs_subst(VetSmallInt v, int pos, int novovalor);
```

Ela recebe como parâmetros um vetor de pequenos inteiros (`v`) um inteiro que indica qual o pequeno inteiro a ser substituído (`pos`, que pode assumir valores de 0 a 3) e um inteiro cujo valor deve ser colocado nessa posição (`novovalor`).

3. Traduza as funções `foo` e `boo` abaixo para assembly IA-32 (o assembly visto em sala), utilizando as regras usuais de alinhamento, passagem de parâmetros, salvamento de registradores e resultados em C/linux. Traduza o mais diretamente possível o código de C para assembly.

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

   Comente seu código!

   (a) (2,5 pontos)

```
void foo (int *v1, int *v2, int n) {
  int a;
  int i;
  for (i=0;i<n;i++) {
    a = f(i);
    if ((a>0) && (a<n))
      v1[i] = v2[a];
  }
  return s;
}
```

   (b) (2,0 pontos)

```
struct x {
  int v1;
  struct x *next;
};

int boo (struct x *px, int val) {
  while (px != NULL) {
    if (px->v1 == val) return 1;
    px = px->next;
  }
  return 0;
}
```

Boa Prova!