PUC-Rio – Software Básico – INF1018
Prova 1 – Turma 3WA – 04/05/2017

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 elem {
   short s1;
   int *s2;
   char c;
   int i;
};
int main(void) {
   struct elem elems[2];
   elems[0].s1 = -4;
   elems[0].s2 = &elems[1].i;
   elems[0].c = 'c';
   elems[0].i = 1030 << 2;
   dump(elems, sizeof(struct elem));
   return 0;
}
```

   Descreva o que este programa irá imprimir quando executado, *justificando* os valores exibidos mostrando as contas e outras informações usadas para chegar ao resultado. Suponha que a máquina de execução seja *little-endian* com as convenções de alinhamento do Linux no IA-64 vistas em sala. Além disto considere que `elems` seja alocado na posição de memória 0x7F00006DA260 e que o valor do caracter 'a' na tabela ASCII seja 97 (decimal). Coloque **PP** nas posições correspondentes a *padding*.

2. (2,5 pontos) Escreva em C uma função que dado um inteiro contendo o *Code Point* de um caractere UNICODE na faixa de U+0800 a U+FFFF, retorne o caractere codificado em UTF-8 num vetor de char:

```
void i2utf8(int codePoint, char *saidaUtf8);
```

   Para sua referência, a tabela abaixo apresenta o número de bytes necessários para representar cada faixa de valores de códigos UNICODE, e a codificação usada para cada uma dessas faixas:

Código UNICODE	Representação em UTF-8
U+0000 - U+007F	0xxxxxxx
U+0080 - U+07FF	110xxxxx 10xxxxxx
U+0800 - U+FFFF	1110xxxx 10xxxxxx 10xxxxxx
U+10000 - U+10FFFF	11110xxx 10xxxxxx 10xxxxxx 10xxxxxx

   Por exemplo, se a entrada for U+2640 (♀), a saída será [0xE2,0x99, 0x80]. Considere que o espaço necessário para a saída em saidaUft8 já foi alocado.

3. Traduza as funções `boo` e `foo` abaixo para assembly IA-64 (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)

```
inf ops(int i);

int boo(int a[], int n)
{
   int r = 0; int i = 0;
   while (i < n)
   {
      if (ops(a[i]))
         r = r + a[i];
      i++;
   }
   return r;
}
```

   (b) (2,5 pontos)

```
struct No
{
   int info;
   struct No* esq;
   struct No* dir;
};

int foo(struct No* v)
{
   if (v == 0)
      return 0;
   else
   {
      int sf = foo(v->esq)+foo(v->dir);
      if (sf == 0)
         return v->info;
      else
         return sf;
   }
}
```

Boa Prova!