

PUC-Rio – Software Básico – INF1018
Prova 1 – Turma 3WA – 17/10/2023

1. (2,0 pontos) Escreva em C uma função que dado um ponteiro para uma cadeia de caracteres no formato UTF-8 retorne a quantidade de bytes usados na codificação do primeiro caractere desta cadeia. Use o seguinte protótipo e assuma que a cadeia recebida sempre possui pelo menos um caractere UTF-8:

```
int numBytesUtf8(char *Utf8Stream);
```

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 cadeia apontada por `Utf8Stream` for `{0x50, 0xD1, 0x90, 0x33, 0x0A, 0x00}` então o retorno da função `numBytesUtf8` será 1. Ou, se a cadeia apontada por `Utf8Stream` for `{0xD1, 0x90, 0x33, 0x0A, 0x00}` o retorno da função será 2.

2. (3,0 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 dict;
    int *p;
    char c;
};

int v[] = {7, 8, -1, 3};
struct X x = {-10, &tams[2], 'c'};

int main (void) {
    dump (&x, sizeof(X));
    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 `v` seja alocado na posição de memória `0x60c010`, que o valor do caractere 'a' na tabela ASCII seja 97 (decimal) e que `x` seja alocado na posição de memória `0x3ec020`. Coloque **PP** nas posições correspondentes a *padding*.

3. Traduza as funções `acerta` e `boo` 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.

Comente seu código!

(a) (2,5 pontos)

```
int acerta (int u[], int i)
{
    return u[i];
}
```

(b) (2,5 pontos)

```
struct S
{
    int v;
    struct S *prox;
};

extern int norma[];

int boo(struct S *s, int n)
{
    int acum = 0;

    while (s)
    {
        s->v = acerta(norma, n+acum);
        acum += n;
        s = s->prox;
    }
    return acum;
}
```

Boa Prova!