

PUC-Rio – Software Básico – INF1018
Prova 1 – Turma 3WA – 06/10/2022

1. (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 S {
    int v[2];
    char c;
    int *p;
};
int main(void) {
    struct S s;
    s.v[0] = 1025;
    s.v[1] = -17;
    s.c = 'g';
    s.p = &s.v[1];
    dump(&s, sizeof(struct S));
    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 *s* seja alocado na posição de memória 0x7ffcf2439e10 e que o valor do caracter 'a' na tabela ASCII seja 97 (decimal). Coloque **PP** nas posições correspondentes a *padding*.

2. (2,0 pontos) Implemente uma função em C, que dado um inteiro sem sinal retorne um inteiro sem sinal onde a ordem dos 4 bytes do parâmetro de entrada foi invertida. Use o seguinte protótipo:

```
unsigned int inverteOrdemBytes(unsigned int i);
```

Por exemplo, se a entrada *i* for igual a 0x01020304 a saída devera ser 0x04030201.

3. (5,0 pontos) Traduza a função `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.

Comente seu código!

```
(a) int gera(char c1, char c2);

int foo(int n, char c, char *v)
{
    int i, s = 0;

    for (i=0; i<n; i++)
    {
        if (v[i] >= c)
            s += gera(v[i], c);
    }
    return s;
}
```

Boa Prova!