# Exercícios de codificação Assembly

**Questão 1.** Implemente um código Assembly no processador hipotético Neander que seja capaz de efetuar uma subtração. As posições de memória 128 e 129 devem conter, respectivamente, o minuendo e o subtraendo. O resultado deverá ser armazenado na posição 130.

Casos de teste para validação do programa desenvolvido:

| Endereço | 128 | 129 | 130 |
|---|---|---|---|
| **Caso 1** | 5 | 2 | 3 |
| **Caso 2** | 15 | 25 | 246 (-10 em complemento de 2)<br>11110110 |
| **Caso 3** | 55 | 70 | 241 (-15 em complemento de 2)<br>11110001 |
| **Caso 4** | 126 | -1 (não esqueça de representar em complemento de 2. Considerando que o Neander não entende na digitação o complemento de 2 insira o número -1 convertendo o binário resultante do complemento de dois para o decimal.) | 127 |

**Questão 2.** Implemente um código que seja capaz de calcular o endereço da rede a qual um IPv4 pertence dados o IPv4 e a Máscara. A memória de dados deve ser organizada da seguinte forma:

Palavras 128, 129, 130 e 131 – Bytes do IPv4
Palavras 132, 133, 134, 135 – Bytes da Máscara
Palavras 136, 137, 138, 139 – IP da rede calculado

Casos de teste para validação do programa desenvolvido:

| Endereço | IPv4 | Máscara | IP da rede calculado |
|---|---|---|---|
| **Caso 1** | 10.10.10.1 | 255.0.0.0 | 10.0.0.0 |
| **Caso 2** | 192.168.1.35 | 255.255.255.0 | 192.168.1.0 |
| **Caso 3** | 200.132.35.123 | 255.255.255.192 | 200.132.35.64 |

**Questão 3.** Implemente um código Assembly no Neander que seja capaz de contar até um número previamente estabelecido na posição de memória 129. A posição de memória 128 deve conter inicialmente o valor zero (0) e ir sendo atualizada conforme o contador for sendo incrementado durante a execução do programa.

**Questão 4.** Implemente um código Assembly no Neander que seja capaz de decrementar um valor até zero (0) a partir de um número previamente estabelecido na posição de memória 129. A posição de memória 128 deve conter inicialmente o valor informado na posição 129 e ir sendo decrementado até chegar a zero (0) durante a execução do programa.

**Questão 5:** Escrever um programa para o simulador Neander que multiplique dois valores inteiros positivos de 8 bits e armazene o produto obtido (8 bits). Não é necessário se preocupar com números negativos e multiplicações que provoquem overflow.. Os valores dados e calculados devem estar obrigatoriamente nas seguintes posições:

Palavra 128 - multiplicando
Palavra 129 - multiplicador
Palavra 130 – resultado da multiplicação

Casos de teste para validação do programa desenvolvido:

| Endereço | 128 | 129 | 130 |
|---|---|---|---|
| **Caso 1** | 5 | 3 | 15 |
| **Caso 2** | 23 | 10 | 230 |
| **Caso 3** | 1 | 200 | 200 |