



## 1. Introdução

Pretende-se com esta aula prática que os alunos entendam o código gerado por um compilador de C para um programa apenas com variáveis escalares e estruturas condicionais, utilizando diferentes níveis de optimização. Para atingir este objectivo o aluno deve desenvolver um pequeno programa em C, compilá-lo, interpretar o código gerado e alterá-lo.

## 2. Linguagem de alto nível

Escreva em C, usando o editor de texto que considerar mais adequado, o seguinte programa:

```
prog.c
#include <stdio.h>
int i=10,j=5,k,l;
int main ()
{
    i =10;
    j = 5;
    if (i<j)    k = i+j;
    else       k = i-j;
    l = k*3;

    printf ("i=%d\tj=%d\n", i, j);
    printf ("k=%d\tl=%d\n", k, l);
}
```

## 3. Compilação sem optimização

Compile o programa `prog.c` usando o comando

```
gcc -S prog.c
```

Analisando o código *assembly*, responda às seguintes questões:

**Questão 1** – Identifique as instruções responsáveis pelo teste do `if (...)`.

**Questão 2** – Identifique as instruções que implementam as operações `k=i+j` e `l=k*3`.

**Questão 3** – Explique o porquê da instrução `jmp .L3`?

**Questão 4** – Modifique o código *assembly* de forma que `l = k*4`. Verifique o resultado gerando o programa final (`gcc -o prog prog.s`) e executando-o.

#### 4. Compilação com otimização O1

Recompile o programa `prog.c` usando o comando

```
gcc -O1 -S prog.c
```

Analisando o código *assembly*, responda às seguintes questões:

**Questão 5** – Consegue localizar o código associado ao teste condicional `if (...)`? Como explica esta situação?

**Questão 6** – Identifique e explique as instruções associadas à instrução `l = k*3`.

#### 5. Compilação com otimização O2

Recompile o programa `prog.c` usando o comando

```
gcc -O2 -S prog.c
```

Analisando o código *assembly*, responda à seguinte questão:

**Questão 7** – Identifique e explique as instruções associadas ao cálculo dos valores de `k` e `l`. Como explica esta situação?