# ISA do IA-32 (inc. funções)

## Teste 3

| **Nº** | **Nome** |
|---|---|

**1.** Considere a estrutura de controlo de um ciclo iterativo, num programa escrito numa linguagem imperativa (por exemplo, em C a estrutura *`for(i=0;i<max;i++)`*). Quando o compilador tiver de gerar código para implementar esta estrutura em linguagem máquina dum processador do tipo do IA-32 – onde as únicas instruções disponíveis são um salto incondicional (do tipo *`goto`*) e um salto condicional (do tipo *`if<cond>then`*) – **mostre** como é que o compilador transforma o código deste ciclo para que ele possa ser traduzido para linguagem máquina.

**2.** Considere a seguinte expressão aritmética com inteiros, no corpo de uma função em C, para execução num PC (*little endian*),

`vec[i]=b+c`

em que `vec` é a única variável local que se encontra em memória, (um *array* definido com a dimensão `20`), `i` é o índice do *array* que foi passado como 2º argumento da função (o 1º foi um valor real de precisão dupla), e `b` e `c` são variáveis locais (nos registos `%ecx` e `%esi`). Os registos `%eax`, `%ebx` e `%edx` são usados na função para conterem, respectivamente, os valores de retorno da função, de `i`, e de apontador para o início do *array*.

**Estimar a dimensão mínima**, em *bytes*, que a estrutura da *stack frame* desta função ocupará na pilha (na memória do computador), quando a função for invocada e executada. Justifique os cálculos.
Nota: não esquecer que é na *stack frame* que ficam os argumentos passados para a função, se salvaguardam os valores dos registos (apenas os necessários, e indicar quais) e se armazenam as variáveis locais que não podem ser alocadas a registos.

**3.** Considere os pressupostos do exercício anterior.

**Escreva** a instrução em *assembly* que copia o valor do argumento contendo o índice do *array* para o registo **%edx**.