Execução de Programas

Considere os seguintes programas em C:

Programa #1

```
#define VALUE 10

int main() {
   int i, mult_table[10];

   for (i = 0; i < 10; i++) {
      mult_table[i] = i * VALUE;
   }
   return 0;
}
```

Programa #2

```
#include <stdio.h>

#define VALUE 10

int main() {
   int i;

   for (i = 0; i < 10; i++) {
     printf("%d x %d = %d\n", i, VALUE, i * VALUE);
   }
   return 0;
}
```

Programa #3

```
#include <stdio.h>

#define STRING_MAX_SIZE 100
#define PWD "MyPassword"

int main(){
   char s[STRING_MAX_SIZE];

#ifdef DEBUG
   printf("Write text to encrypt: ");
#endif
   scanf("%s", s);
   printf("crypt(%s) = %s\n", s, crypt(s, PWD));
#ifdef DEBUG
   printf("Done!\n");
#endif
   return 0;
}
```

1. Compile cada um dos 3 programas com o **gcc** utilizando a opção **-Wall**. No caso do programa #3, compile ainda com e sem a opção **-DDEBUG**. Identifique o ponto do código relativo a eventuais avisos de compilação e, caso obtenha erros de compilação, altere o programa e/ou as opções do **gcc** para que a compilação seja bem sucedida. Execute cada um dos programas e verifique o seu output.

2. Para cada programa, verifique quais são as alterações realizadas no passo de pré-processamento do compilador através da invocação do comando **gcc -E progX.c > progX.i** (ou invocando diretamente o pré-processador através do

comando **cpp progX.c > progX.i**). Repita o pré-processamento para o programa #3, mas desta vez adicione a opção **-DDEBUG**.

3. Para cada programa, gere o código assembly através da invocação do comando **gcc -S progX.i** (output em **progX.s**) e compare-o com o código resultante do pré-processamento. Tente identificar as estruturas principais do programa: ciclos, saltos, saltos condicionais, chamadas de funções.

4. Para cada programa, gere o código assembly com a opção adicional de otimização de código **-O2** e compare as diferenças entre as versões com e sem optimização.

5. Para cada programa, compile o código assembly para código binário (objeto) através da invocação do comando **gcc -c progX.s** (output em **progX.o**). Use o comando **nm progX.o** para verificar quais os símbolos existentes (T) e respetivo endereço de memória e quais os símbolos não definidos (U) no ficheiro objeto. Use ainda o comando **objdump -d progX.o** para ver o conteúdo do ficheiro objeto.

6. Para cada programa, compile o ficheiro objeto para código executável através da invocação do comando **gcc progX.o -o progX.out** e em seguida execute o comando **strip progX.out -o progX.sout**. Use novamente os comandos **nm** e **objdump -d** agora sobre o código executável em **progX.out** e **progX.sout**. Use também o comando **ldd progX.out** para verificar as dependências do código executável.

7. Execute o comando **file** sobre cada um dos ficheiros **progX.c**, **progX.i**, **progX.s**, **progX.o**, **progX.out** e **progX.sout** e identifique a arquitetura do código gerado pelo compilador.

8. (saber mais) Aceda à plataforma Compiler Explorer e explore a possibilidade de gerar código assembly para uma diversidade de diferentes tipos de arquiteturas. Em particular, veja o código assembly para as arquitecturas MIPS.