Trabalho Prático Nº 2
Organização da memória em segmentos; Modos de endereçamento
Utilização do Turbo Pascal com linguagem assembly 8086 (inline assembler)
Instruções do 8086: MOV, LEA, LES, LDS ADD, MUL, PUSH, POP, MOVSB, MOVSW
Acesso a variáveis e estruturas de dados

- **Organização da memória em segmentos; Modos de endereçamento**

1) Supondo que CS=1000h, DS=1200h, SS=0F00h, ES=14F0h, IP=0010h, BX=120Eh, BP=340Fh, SI=A000h e DI=3000h.
a) Determine o endereço em memória (endereço físico) da próxima instrução a ser executada.
b) Determine o endereço efectivo e o endereço em memória dos seguintes operandos:

[2000h]	ES:[2000h]	
[BX]	[BP]	ES:[BX]
[BX][2000h]	[BX+2]	ES:[BX][40h]
[BP][2000h]	[BP+10]	ES:[BP][40h]
[SI]	[DI]	ES:[SI]
[SI][0100h]	[DI+20]	ES:[SI+10h]
[BX][SI]	[BX+DI]	ES:[BX+SI]
[BX][SI][0100h]	[BX+DI+32]	ES:[BX+SI+32]

- **Utilização do Turbo Pascal com linguagem assembly (8086) (inline assembler) Instruções do 8086: MOV, LEA, LES, LDS, ADD, MUL, PUSH, POP, MOVSB, MOVSW**
- **Acesso a variáveis e estruturas de dados**

O "inline assembler" permite a utilização de linguagem assembly directamente em programas escritos e compilados no ambiente de desenvolvimento do Turbo Pascal. Para o efeito, as instruções assembly devem ser delimitadas pelas palavras reservadas asm .. end, conforme se indica no exemplo seguinte.

```
Program teste;
Var x,y:Integer;
Begin
......{Instruções em Pascal}
asm
......{Instruções em Assembly}
end;
......{Instruções em Pascal}
End.
```

Recorrendo a programas com a estrutura anterior, poderemos manipular objectos (variáveis, constantes, estruturas de dados, etc..) utilizando a linguagem Pascal ou a linguagem Assembly.
Num bloco de instruções assembly, não poderão ser utilizados procedimentos ou funções pré-definidas, como, por exemplo, os procedimentos write(ln) e read(ln).
Os identificadores utilizados na declaração de constantes e variáveis em linguagem Pascal podem ser utilizados na linguagem assembly. Por exemplo,

```
.....
Const cte=20;
Var x,y:Integer;
```

```
.....
begin
....
asm
MOV AX, x {Copia o valor da variável x para o registo AX}
MOV AX, cte {Copia o valor 20 para o registo AX}
end;
....
```

Na fase de compilação do programa, cada variável é associada a um endereço de memória onde esta é armazenada (DS: [Deslocamento(offset)]. No exemplo anterior, supondo que, à variável x, o compilador atribuiu a posição de memória [0044h] no segmento de dados DS, a instrução

```
MOV AX, x
```

é equivalente a

```
MOV AX, [0044h] {endereçamento directo}.
```

No caso da constante cte, a instrução

```
MOV AX, cte
```

é equivalente a

```
MOV AX, 20 {endereçamento imediato}.
```

Para a detecção e correcção de erros, a execução do programa pode ser acompanhada passo a passo, permitindo, simultaneamente, visualizar o conteúdo dos registos do CPU.

2)- Considere as seguintes declarações em Pascal:

```
const cte=2;
var x,y : Byte;
    xx,yy,zz : Integer;
    logico : Boolean;
```

Utilizando o "inline assembler", escreva um programa que efectue as operações equivalentes às instruções em Pascal:

a)
```
x:=24;
y:=x;
```

b)
```
x:=32;
y:=x+cte;
```

c)
```
xx:=100;
yy:=200;
xx:=xx+yy;
```

e)
```
logico:=True;
```

d)
```
xx:=10;
yy:=20;
zz:=xx;
xx:=yy;
yy:=zz;
```

3) Considere as seguintes declarações em Pascal:

```
var   tabbyte: array[0..2] of byte;
      tabint: array[0..2] of integer;
      i,j:word;
```

Utilizando o "inline assembler", escreva um programa que efectue as operações equivalentes às instruções em Pascal:

a)
```
tabbyte[0]:=15;
tabbyte[1]:= tabbyte[0]+2;
tabbyte[2]:= tabbyte[1]+2;
```

b)
```
i:=0; j:=2;
tabint[i]:=15;
tabint[i]:= tabint[i]+10;
tabint[j]:= tabint[i]+20;
```

4) Considere as seguintes declarações em Pascal:
```
var   tabint: array[0..2,0..3] of Integer;
      i,j:Word;
```

Utilizando o "inline assembler", escreva um programa que efectue as operações equivalentes às instruções em Pascal:

a)
```
tabint[0,1]:=15;
tabint[1,2]:=20;
```

b)
```
i:=2;j:=1;
tabint[i,j]:=50;
```

5)- Considere as seguintes declarações em Pascal:
```
var s: string[2];
```
Utilizando o "inline assembler", escreva um programa que efectue as operações equivalentes às instruções em Pascal:

a) `s:='a';` b) `s:='AB';` c) `s:='';`

6) Considere as seguintes declarações em Pascal:
```
var   rec:record
          i:integer;
          s:string[2];
          b:byte;
          c:char;
          end;
```

Utilizando o "inline assembler", escreva um programa que efectue as operações equivalentes às instruções em Pascal:
```
rec.i:=15;
rec.s:='AB';
rec.b:=25;
rec.c:='C';
```

7) Considere as seguintes declarações em Pascal:
```
type rectype=record
          i:integer;
          c:char;
          end;
var tabrec:array[0..3] Of rectype;
    i,j:word;
```

Utilizando o "inline assembler", escreva um programa que efectue as operações equivalentes às instruções em Pascal:

```
i:=0;j:=2;
tabRec[i].i:=15;
tabRec[i].c:='A';
tabRec[j].i:=25;
tabRec[j].c:='B';
```

8) Considere as seguintes declarações em Pascal:

```
type rectype=record
          c:char;
          i:integer;
          end;
var t1,t2:array[0..3] of rectype;
    i,j:word;
```

Supondo que a variável T2 é incicializada com as seguintes instruções:

```
For i:=0 to 3 do
   begin
   t2[i].i:=10+i;
   t2[i].c:=chr(ord('A')+i);
   end;
```

Utilizando o "inline assembler", escreva um programa que efectue as operações equivalentes às instruções em Pascal:

a)
```
j:=2;i:=1;
t1[j]:=t2[i];
```

b)
```
t1:=t2;
```

9) Considere as seguintes declarações em Pascal:

```
type ptrrec=^rectype;
     rectype=record
          i:integer;
          c:char;
          end;
var pt:ptrrec;
```

Supondo que a variável pt é incicializada com as seguintes instruções:

```
new(pt);
pt^.i:=20;
pt^.c:='E';
```

Utilizando o "inline assembler", escreva um programa que efectue as operações equivalentes às instruções em Pascal:

```
pt^.i:=pt^.i+10;
pt^.c='A';
```