Physikalisch-Technische Lehranstalt Wedel

Staatsexamen für Technische Assistenten SoSe2004
schriftliche Prüfung im Fach Prozesstechnik
gewählter Vorschlag

Dauer : 180 Minuten **keine externen Hilfsmittel**

Aufgabe :

Übersetzen Sie zunächst die TurboPascal-Unit *Tools* in ein Assembler-Programm für eine Stack-Maschine, die nur folgende 16-Bit-Operationen beherrscht :

- Wert auf dem Stack ablegen (PUSH)
- Wert vom Stack holen (POP)
- Unterprogrammaufruf (CALL)
- Unterprogrammrücksprung (RET)
- Auslösen Softwareinterrupt (INT)
- Rücksprung aus Interruptserviceroutine (IRET)
- Wert kopieren (MOV)
- Wert inkrementieren (INC)
- Wert dekrementieren (DEC)
- Test auf Gleich- und Ungleichheit (CMP)
- Sprung bei Gleichheit (JE)
- Sprung bei Ungleichheit (JNE)
- unbedingter Sprung (JMP)

Die Adressierungsarten und Register entsprechen jeweils denen des 8086-Prozessors.

Ziel ist die Einbindung Ihres vom Assembler nach Objektcode übersetzten Assembler-Programms in das TurboPascal-Programm *SoSe2004*.

Übersetzen Sie anschließend auch das TurboPascal-Programm *SoSe2004* in ein Assembler-Programm für eine bereits am Beginn der Aufgabenstellung vom Befehlsumfang her beschriebene Stack-Maschine.

Vergessen Sie nicht die hinreichende Kommentierung Ihrer Assembler-Programme z.B. durch Zuordnung der Pascal-Befehle zu den Assembler-Befehlen.

Zum besseren Verständnis der Klausur ist die Ausgabe des Programms *SoSe2004* beigefügt.

Das PTL-Team wünscht viel Erfolg

```
Unit Tools;

Interface

Function Addup(X,Y:Word):Word;
Procedure Subtract(X,Y:Word;Var Z:Word);
Function Multiply(X,Y:Word):Word;
Procedure Divide(X,Y:Word;Var Q,R:Word);
Procedure Character(X:Word);
Procedure Binaer(X,Y:Word);

Implementation

Function GreaterEqual(X,Y:Word):Boolean;

Begin
  If (X=0) And (Y=0) Then
    GreaterEqual := True
  Else
    If (Y=0) Then
      GreaterEqual := True
    Else
      If (X=0) Then
        GreaterEqual := False
      Else
        GreaterEqual := GreaterEqual(X-1,Y-1)
End;

Function Addup(X,Y:Word):Word;

Begin
  If Y = 0 Then Addup := X
           Else Addup := Addup(X,Y-1) + 1
End;

Procedure Subtract(X,Y:Word;Var Z:Word);

Begin
  If Y = 0 Then Z := X
           Else Begin
                Subtract(X,Y-1,Z);
                Z := Z-1
                End
End;

Function Multiply(X,Y:Word):Word;

Begin
  Case Y Of
    0  : Multiply := 0;
    1  : Multiply := X;
    Else Multiply := Addup(X,Multiply(X,Y-1))
  End
End;

Procedure Divide(X,Y:Word;Var Q,R:Word);

Var Z : Word;

Begin
  If GreaterEqual(X,Y) Then Begin
    Subtract(X,Y,Z);
    Divide(Z,Y,Q,R);
    Q := Q+1
  End Else Begin
    Q := 0;
    R := X
  End
End;
```

```
Procedure Character(X:Word);

Begin
  Asm
    MOV AX,0200h
    MOV DX,X
    INT 21h
  End
End;

Procedure Binaer(X,Y:Word);

Var B,Q,R : Word;

Begin
  B := X;
  If Y <> 0 Then Begin
    Divide(X,2,Q,R);
    B := R;
    Binaer(Q,Y-1)
  End;
  If B = 0 Then Character(Ord('0'))
           Else Character(Ord('1'))
End;

End.
```

```
Program SoSe2004;

{$IFDEF PAS}
Uses Tools;
{$ENDIF}

{$IFDEF ASM}
{$L Tools}
Function Addup(X,Y:Word):Word; Far; External;
Procedure Subtract(X,Y:Word;Var Z:Word); Far; External;
Function Multiply(X,Y:Word):Word; Far; External;
Procedure Divide(X,Y:Word;Var Q,R:Word); Far; External;
Procedure Character(X:Word); Far; External;
Procedure Binaer(X,Y:Word); Far; External;
{$ENDIF}

Var I,J,Q,R,S : Word;

Begin
  For I := 1 To 10 Do
    For J := 1 To I Do Begin
      Binaer(I,8);
      Character(Ord(','));
      Binaer(J,8);
      Character(Ord(':'));
      Binaer(Addup(I,J),8);
      Character(Ord(';'));
      Subtract(I,J,S);
      Binaer(S,8);
      Character(Ord(';'));
      Binaer(Multiply(I,J),8);
      Character(Ord(';'));
      Divide(I,J,Q,R);
      Binaer(Q,8);
      Character(Ord(','));
      Binaer(R,8);
      Character(13);
      Character(10)
    End
End.
```

000000001,000000001:000000010;000000000;000000001;000000001,000000000
000000010,000000001:000000011;000000001;000000010;000000010,000000000
000000010,000000010:000000100;000000000;000000100;000000001,000000000
000000011,000000001:000000100;000000010;000000011;000000011,000000000
000000011,000000010:000000101;000000001;000000110;000000001,000000001
000000011,000000011:000000110;000000000;000001001;000000001,000000000
000000100,000000001:000000101;000000011;000000100;000000100,000000000
000000100,000000010:000000110;000000010;000001000;000000010,000000000
000000100,000000011:000000111;000000001;000001100;000000001,000000001
000000100,000000100:000001000;000000000;000010000;000000001,000000000
000000101,000000001:000000110;000000100;000000101;000000101,000000000
000000101,000000010:000000111;000000011;000001010;000000010,000000001
000000101,000000011:000001000;000000010;000001111;000000001,000000010
000000101,000000100:000001001;000000001;000010100;000000001,000000001
000000101,000000101:000001010;000000000;000011001;000000001,000000000
000000110,000000001:000000111;000000101;000000110;000000110,000000000
000000110,000000010:000001000;000000100;000001100;000000011,000000000
000000110,000000011:000001001;000000011;000010010;000000010,000000000
000000110,000000100:000001010;000000010;000011000;000000001,000000010
000000110,000000101:000001011;000000001;000011110;000000001,000000001
000000110,000000110:000001100;000000000;000100100;000000001,000000000
000000111,000000001:000001000;000000110;000000111;000000111,000000000
000000111,000000010:000001001;000000101;000001110;000000011,000000001
000000111,000000011:000001010;000000100;000010101;000000010,000000001
000000111,000000100:000001011;000000011;000011100;000000001,000000011
000000111,000000101:000001100;000000010;000100011;000000001,000000010
000000111,000000110:000001101;000000001;000101010;000000001,000000001
000000111,000000111:000001110;000000000;000110001;000000001,000000000
000001000,000000001:000001001;000000111;000001000;000001000,000000000
000001000,000000010:000001010;000000110;000010000;000000100,000000000
000001000,000000011:000001011;000000101;000011000;000000010,000000010
000001000,000000100:000001100;000000100;000100000;000000010,000000000
000001000,000000101:000001101;000000011;000101000;000000001,000000011
000001000,000000110:000001110;000000010;000110000;000000001,000000010
000001000,000000111:000001111;000000001;000111000;000000001,000000001
000001000,000001000:000010000;000000000;001000000;000000001,000000000
000001001,000000001:000001010;000001000;000001001;000001001,000000000
000001001,000000010:000001011;000000111;000010010;000000100,000000001
000001001,000000011:000001100;000000110;000011011;000000011,000000000
000001001,000000100:000001101;000000101;000100100;000000010,000000001
000001001,000000101:000001110;000000100;000101101;000000001,000000100
000001001,000000110:000001111;000000011;000110110;000000001,000000011
000001001,000000111:000010000;000000010;000111111;000000001,000000010
000001001,000001000:000010001;000000001;001001000;000000001,000000001
000001001,000001001:000010010;000000000;001010001;000000001,000000000
000001010,000000001:000001011;000001001;000001010;000001010,000000000
000001010,000000010:000001100;000001000;000010100;000000101,000000000
000001010,000000011:000001101;000000111;000011110;000000011,000000001
000001010,000000100:000001110;000000110;000101000;000000010,000000010
000001010,000000101:000001111;000000101;000110010;000000010,000000000
000001010,000000110:000010000;000000100;000111100;000000001,000000100
000001010,000000111:000010001;000000011;001000110;000000001,000000011
000001010,000001000:000010010;000000010;001010000;000000001,000000010
000001010,000001001:000010011;000000001;001011010;000000001,000000001
000001010,000001010:000010100;000000000;001100100;000000001,000000000