به نام خدا

تمرین چهارم درس ریزپردازنده

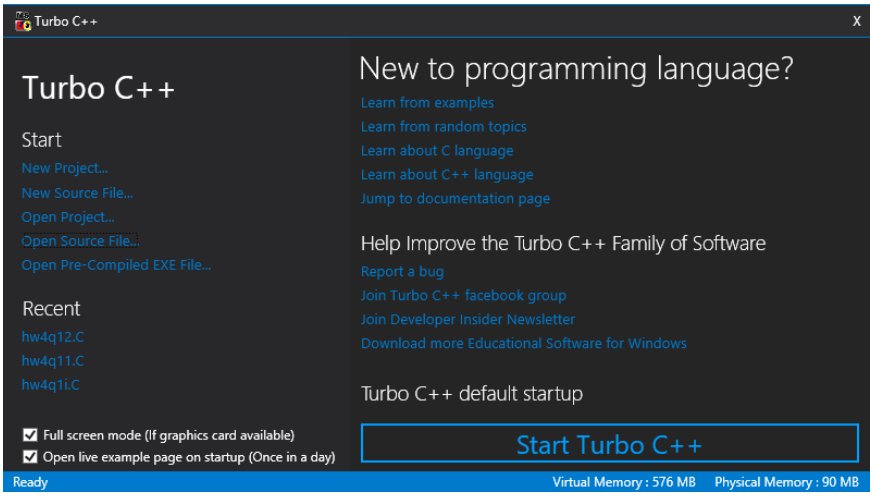
غزل زمانی‌نژاد

۹۷۵۲۲۱۶۶

1. int86 برای دسترسی به وقفه های DOS وBIOS وجود دارد. مقادیر موجود در رجیسترهای CPU را به ساختاری میبرد که در آن متغیرها معادل رجیسترهای CPU است.

Intdos برای صدا کردن وقفه های DOS است. آرگومان آن رجیستر هایی غیر از AX,DL هستند و یا با ست کردن carry flag خطا را نشان میدهند.

در این سوال ابتدا توربو++ را نصب میکنیم.



برای پیاده سازی مشابه کتاب ص 180 عمل میکنیم. تاریخ روز را دریافت میکنیم آن را با یک تابع از میلادی به شمسی تبدیل میکنیم و آن را در خروجی نمایش میدهیم.

```
main()
{
    long year;
    long month;
    long day;
    int result[2];
    union REGS regin,regout;
    regin.h.ah = 0x2A;
    int86(0x21,&regin, &regout);
    day = (long) regout.h.dl;
    month = (long) regout.h.dh;
    year = (long) regout.x.cx;
    convert_data(year, month, day, result);
    printf("data: %d/%d/%d\n", result[0], result[1], result[2]);
    return 0;
}
```

برای پیاده سازی تابع تبدیل تاریخ از سایت استفاده شده است.

```
int *convert_data(long gy, long gm, long gd, int result[]) {
  long jy, gy2, days, g_d_m[12] = {0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334};
  gy2 = (gm > 2) ? (gy + 1) : gy;
  days = 355666 + (365 * gy) + ((int)((gy2 + 3) / 4)) - ((int)((gy2 + 99) / 100)) + ((int)((gy2 + 399) / 400)) + gd + g_d_m[gm - 1];
  jy = -1595 + (33 * ((int)(days / 12053)));
  days %= 12053;
  jy += 4 * ((int)(days / 1461));
  days %= 1461;
  if (days > 365) {
    jy += (int)((days - 1) / 365);
    days = (days - 1) % 365;
  }
  result[0] =(int) jy;
  if (days < 186) {
    result[1]/*jm*/ = 1 + (int)(days / 31);
    result[2]/*jd*/ = 1 + (days % 31);
  } else {
    result[1]/*jm*/ = 7 + (int)((days - 186) / 30);
    result[2]/*jd*/ = 1 + ((days - 186) % 30);
  }
  return result;
}
```

برنامه را با استفاده از توربو ++ کامپایل و اجرا میکنیم. نتیجه در خروجی مطابق زیر است:

```
C:\TURBOC3\BIN>TC C:\TURBOC3\Projects\hw4q11.C
data: 1400/9/28
data: 1400/9/28
data: 1400/9/28
data: 1400/9/28

Type EXIT to return to Turbo C++. . .
```

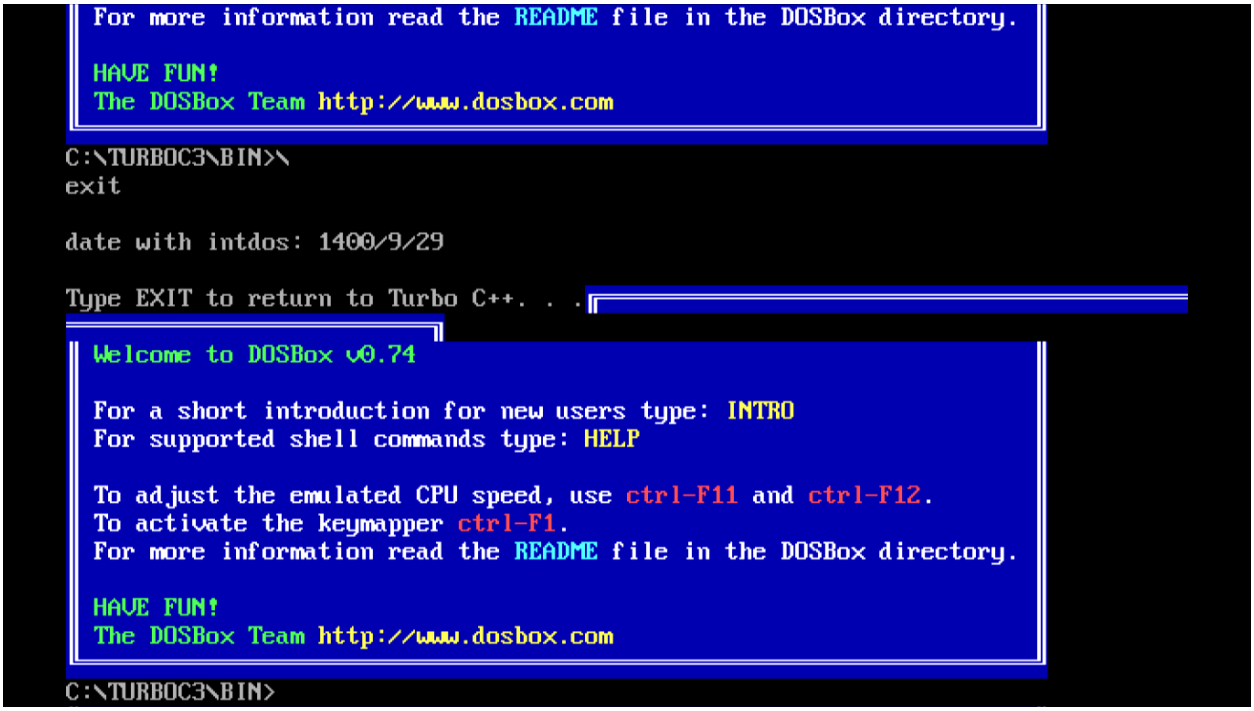
```
Welcome to DOSBox v0.74

For a short introduction for new users type: INTRO
For supported shell commands type: HELP

To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.
To activate the keymapper ctrl-F1.
For more information read the README file in the DOSBox directory.

HAVE FUN!
The DOSBox Team http://www.dosbox.com
```

```
C:\TURBOC3\BIN>_
```

همین کار را با استفاده از وقفه intdos تکرار میکنیم.

```c
main()
{
    long year;
    long month;
    long day;
    int result[2];
    union REGS regin,regout;
    regin.h.ah = 0x2A;
    intdos(&regin, &regout);
    day = (long) regout.h.dl;
    month = (long) regout.h.dh;
    year = (long) regout.x.cx;
    convert_date(year, month, day, result);
    printf("today's result is %d/%d/%d\n", result[0], result[1], result[2]);
    return 0;
}
```

نتیجه:



2. در این کد در بخش data segment متغیرهایی که به آنها نیاز داریم نگه داری میکنیم.

```
.MODEL SMALL
.STACK 64
.DATA
MSG DB "ENTER NUMBER: $"
MSG2  DB      'CHOOSE A NUMBER FOR:',10,13,
      DB      'SUM (0)',10,13,
      DB      'SUB (1)',10,13,
      DB      'MUL (2)',10,13,
      DB      'DIV (3)',10,13,
      DB      'QUIT (4)',10,13,'$'
NUM1 DB 0
NUM2 DB 0
VALUE DB 0
RES  DW ?
```

در تابع MAIN ابتدا با اعمال وقفه یک عدد چند رقمی از کاربر دریافت میکنیم.

```
    ;;;;;;;;;;;;;;;;
    ;GET FIRST NUMBER FROM KEYBOARD
    MOV AH, 9
    LEA DX, MSG
    INT 21H

READ: MOV AH, 1
    INT 21H

    CMP AL, 13
    JE ENDOFNUMBER

    MOV VALUE, AL
    SUB VALUE, 48

    MOV AL, NUM1
    MOV BL, 10
    MUL BL

    ADD AL, VALUE

    MOV NUM1, AL

    JMP READ

    ENDOFNUMBER:
    ;;;;;;;;;;;;;;;;;;;;;;;;;;
```

سپس کنسول را پاک میکنیم و با همان مراحل بالا عدد دوم را دریافت میکنیم و در num2 ذخیره میکنیم.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;
CALL CLEAR

;GET SECOND NUMBER FROM KEYBOARD
MOV AH, 9
LEA DX, MSG
INT 21H

READ2:
MOV AH, 1
INT 21H

CMP AL, 13
JE ENDOFNUMBER2

MOV VALUE, AL
SUB VALUE, 48

MOV AL, NUM2
MOV BL, 10
MUL BL

ADD AL, VALUE

MOV NUM2, AL

JMP READ2

ENDOFNUMBER2:
;;;;;;;;;;;;;;;;;;;;;;;;;;
```

سپس دوباره کنسول را پاک میکنیم و با نشان دادن شماره مربوط به هر عملیات از کاربر میخواهیم عملیات مد نظرش را انتخاب کند. آن را در Value نگهداری میکنیم.

```
CALL CLEAR
;;;;;;;;;;;;;;;;;;;;;;;;;
;GET OPERATION NUMBER AND STORE IN VALUE
MOV AH, 9
LEA DX, MSG2
INT 21H

MOV     AH,01
INT     21H
MOV     BL,AL
AND     BL,0FH
MOV     VALUE,BL
```

سپس بر اساس مقدار value **به یکی از برچسب ها** jump **میکنیم.**

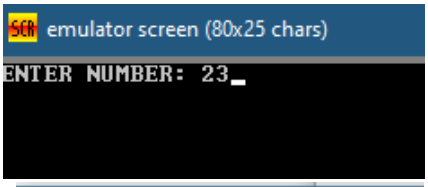
```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;
CALL    CLEAR
;GO TO DIFFERENT LABELS BASED ON VALUE
CMP     VALUE,0
JZ      SUM
CMP     VALUE,01H
JZ      SUBTRACT
CMP     VALUE,02H
JZ      MULTI
CMP     VALUE,03H
JZ      DIVISION
CMP     VALUE,04H
JZ      END
```

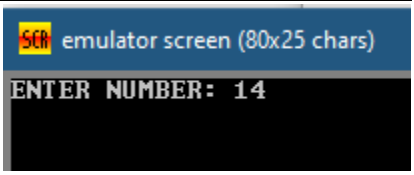
توضیح هریک از لیبل ها:

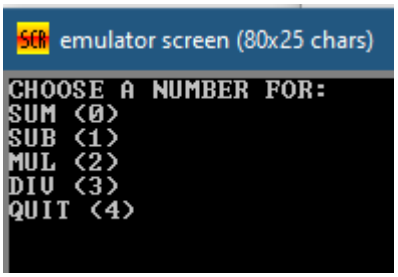
Sum: **عدد اول را در** al **ذخیره میکنیم. عدد دوم را به آن می افزاییم. نتیجه را با استفاده از صدا زدن متد**print **چاپ میکنیم.**

```
SUM:  MOV AH,0
      MOV AL,NUM1
      ADD AL,NUM2
      MOV RES,AX
      CALL PRINT
      JMP END
```

مثال:



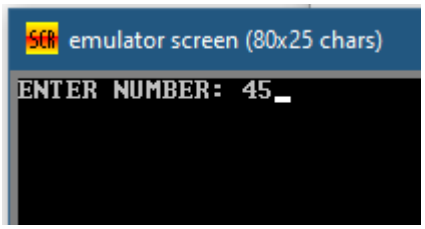


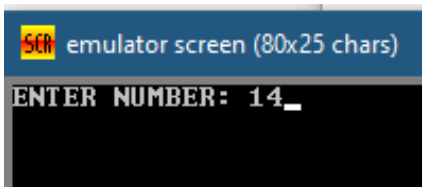


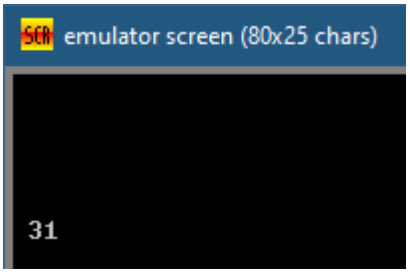
Subtract: عدد اول را در al و عدد دوم را در bl میریزیم. سپس مقایسه میکنیم. در صورتی که عدد دوم بزرگتر باشد به لیبل sub2میرویم و در آنجا بعد از جابه جا کردن دو عدد عملیات تفریق را انجام میدهیم. نتیجه را با صدا زدن print چاپ میکنیم.

```
SUBTRACT:  MOV AH,0
           MOV BH,0
           MOV AL,NUM1
           MOV BL,NUM2
           CMP AL,BL
           JL  SUB2
           SUB AL,BL
           MOV AH,0
           MOV RES,AX
           CALL PRINT
           JMP END

SUB2:      MOV RES,AX
           MOV AL,BL
           MOV BX,RES
           SUB AL,BL
           MOV AH,0
           MOV RES,AX
           CALL PRINT
           JMP END
```

مثال:



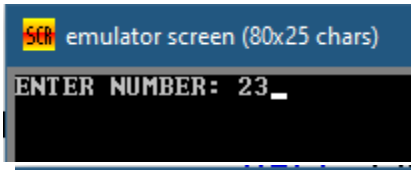


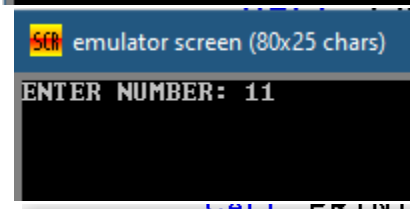


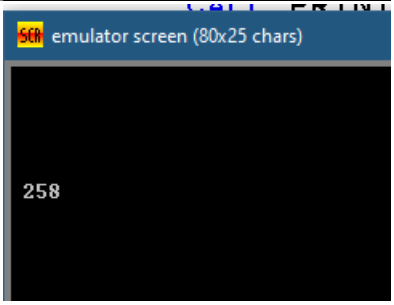
Multi: عدد اول را در al و عدد دوم را در bl میریزیم. با استفاده از دستور mul ضرب را انجام میدهیم. نتیجه را با صدا زدن print چاپ میکنیم.

```
MULTI:     MOV AH,0
           MOV AL,NUM1
           MOV BL,NUM2
           MUL BL
           MOV RES,AX
           CALL PRINT
           JMP END
```

مثال:



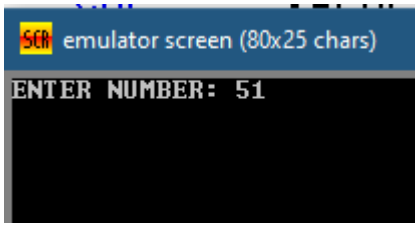


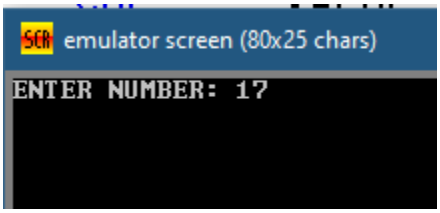


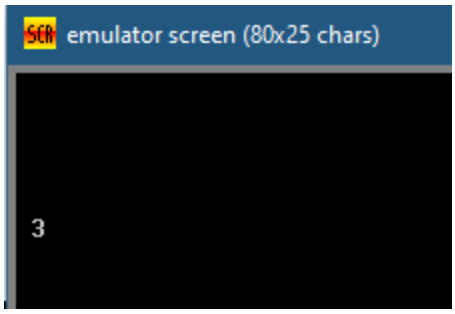
Division: عدد اول را در al و عدد دوم را در bl میریزیم. سپس مقایسه میکنیم. در صورتی که عدد دوم بزرگتر باشد به لیبل div2 میرویم و در آنجا بعد از جابه جا کردن دو عدد عملیات تقسیم را انجام میدهیم. نتیجه را با صدا زدن print چاپ میکنیم.

```
DIVISION:   MOV AH,0
            MOV AL,NUM1
            MOV BL,NUM2
            CMP AL,BL
            JL  DIV2
            DIV BL
            MOV RES,AX
            CALL PRINT
            JMP END

DIV2:       MOV RES,AX
            MOV AL,BL
            MOV BX,RES
            DIV BL
            MOV RES,AX
            CALL PRINT
            JMP END
```

مثال:






SCR
emulator screen (80x25 chars)
3