.MODEL SMALL

newline macro

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

endm

.STACK 100H

.DATA

;welcome page

a1 db 10,13

a2 db 10,13

a3 db 10,13

a4 db 10,13

a5 db 10,13

a6 db 10,13

;choose

a7 db 10,13

a8 db 10,13

a19 db 10,13

a34 db 10,13

a35 db 10,13

a36 db 10,13

a37 db 10,13

a38 db 10,13

a39 db 10,13

;booklist

a9 db 10,13

a10 db 10,13

a11 db 10,13

;english novel list

a12 db '1.Book: THINGS FALL APART : FICTION, Price: $45'

a13 db '2. Book: WUTHERING HEIGHTS, Category: Fiction, Price: $30'

;BANGLA novels list

a21 db '3.ROKTAKTO PRANTOR : FICTION, Price: $45'

a22 db '4.Book: RUDALI : FICTION, Price: $45'

;HISTORIC books list

a27 db '4.THE DIARY OF A YOUNG GIRL:HISTORY, Price: $45'

a28 db '5.ROKTAKTO PRANTOR : HISTORY, Price: $45'

a29 db '6.ROKTAKTO PRANTOR : HISTORY, Price: $45'

a30 db '7.ROKTAKTO PRANTOR : HISTORY, Price: $45'

a31 db '8.ROKTAKTO PRANTOR : HISTORY, Price: $45'

a32 db '9.ROKTAKTO PRANTOR : HISTORY, Price: $45'

a33 db '10.ROKTAKTO PRANTOR : HISTORY, Price: $45'

;DEFINE BOOK PRICES

PRICE1 DW 45

PRICE2 DW 30

PRICE3 DW 45

PRICE4 DW 45

PRICE5 DW 45

PRICE6 DW 45

PRICE7 DW 45

PRICE8 DW 45

PRICE9 DW 45

PRICE10 DW 45

total1 db 'total price:$',

; declare variables here

.CODE

MAIN PROC

; initialize DS

MOV AX,@DATA

MOV DS,AX

; enter your code here

;USE NEWLINE MACRO NEWLINE

newline

newline

;welcome page

mov ah,9

mov dx,offset a1

int 21h

mov dx,offset a2

int 21h

mov dx,offset a3

int 21h

mov dx,offset a4

int 21h

mov dx,offset a5

int 21h

mov dx,offset a6

int 21h

;new line

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

;take input to start

mov ah,9

mov dx,offset a19

int 21h

mov ah,1

int 21h

mov bh,al

sub bh,48

cmp bh,1

je booklist

jmp invalid

booklist:

;newline

mov ah,2

mov dl,10

int 21

mov dl,13

int 21h

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,9

mov dx,offset a7

int 21h

mov dx,offset a9

int 21h

mov dx,offset a11

int 21h

;list choose

mov dx,offset a8

int 21h

mov ah,1

int 21h

mov bh,al

sub bh,48

cmp bh,1

je englishnovels

cmp bh,2

je BANGLAnovels

cmp bh,3

je HISTORICbooks

jmp invalid

;english novels list

englishnovels:

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

;list start

mov ah,9

mov dx,offset a12

int 21h

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,9

mov dx,offset a13

int 21h

;get user selection

mov ah,1

int 21h

mov bl,al

sub bl,48

cmp bl,1

je push\_price1 ;if 1 selected

jmp invalid

push\_price1:

push price1 ;pushing into stack

jmp exit

push\_price2:

push price2 ;p2 into stack

jmp exit

;condition checking

mov dx,offset a34

int 21h

mov ah,1

int 21h

mov bl,al

sub bl,48

jmp invalid

BANGLAnovels:

;newline

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

;list start

mov ah,9

mov dx,offset a28

int 21h

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,9

mov dx,offset a21

int 21h

mov dx,offset a22

int 21h

;get user selection

mov ah,1

int 21h

mov bl,al

sub bl,48

cmp bl,3

je push\_price3

cmp bl,4

je push\_price4

jmp invalid

push\_price3:

push price3

jmp exit

push\_price4:

push price4

jmp exit

;condition checking;

mov dx,offset a34

int 21h

mov ah,1

int 21h

mov bl,al

sub bl,48

jmp invalid

;for exit

mov ah,2

mov dl,0

int 21h

mov dl,13

int 21h

mov ah,9

mov dx,offset a38

int 21h

mov ah,1

int 21h

mov bh,al

cmp bh,1

jmp exit

HISTORICbooks:

;newline

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

;list start

mov ah,9

mov dx,offset a27

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,9

mov dx,offset a28

int 21h

mov dx,offset a29

int 21h

mov dx,offset a30

int 21h

mov dx,offset a31

int 21h

mov dx,offset a32

int 21h

mov dx,offset a33

int 21h

;get user selection

mov ah,1

int 21h

mov bl,al

sub bl,48

cmp bl,5

je push\_price5

cmp bl,6

je push\_price6

cmp bl,7

je push\_price7

cmp bl,8

je push\_price8

cmp bl,9

je push\_price9

cmp bl,10

je push\_price10

jmp invalid

push\_price5:

push price5

jmp exit

push\_price6:

push price6

jmp exit

push\_price7:

push price7

jmp exit

push\_price8:

push price8

jmp exit

push\_price9:

push price9

jmp exit

push\_price10:

push price10

jmp exit

;condition checking

mov dx,offset a34

int 21h

mov ah,1

int 21h

mov bl,al

sub bl,48

mov cx,ax

add ch,48

add cl,48

mov dx,offset a37

mov ah,9

int 21h

mov ah,2

mov dl,ch

int 21h

mov dl,cl

int 21h

mov dl,cl

int 21h

mov dl,'0'

int 21h

mov dl,47

int 21h

mov dl,45

int 21h

;for exit or main menu

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,9

mov dx,offset a38

int 21h

mov ah,9

mov dx,offset a39

int 21h

mov ah,9

mov dx,offset a8

int 21h

mov ah,1

int 21h

sub al,48

cmp al,1

je booklist

cmp al,2

je exit

jmp invalid

mov cx,ax

add ch,48

add cl,48

mov dx,offset a37

mov ah,9

int 21h

mov ah,2

mov dl,ch

int 21h

mov dl,cl

int 21h

mov dl,'0'

int 21h

mov dl,47

int 21h

mov dl,45

int 21h

;for exit

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,9

mov dx,offset a38

int 21h

mov ah,9

mov dx,offset a39

int 21h

mov ah,9

mov dx,offset a8

int 21h

mov ah,1

int 21h

sub al,48

cmp al,1

je booklist

cmp al,2

je exit

jmp invalid

mov bl,20

mov dx,offset a35

mov ah,9

int 21h

mov ah,1

int 21h

sub al,48

mul bl

;aan

mov cx,ax

add ch,48

add cl,48

mov dx,offset a37

mov ah,9

int 21h

mov ah,2

mov dl,ch

int 21h

mov dl,cl

int 21h

mov dl,47

int 21h

mov dl,45

int 21h

;for exit

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,9

mov dx,offset a38

int 21h

mov ah,9

mov dx,offset a39

int 21h

mov ah,9

mov dx,offset a8

int 21h

mov ah,1

int 21h

sub al,48

cmp al,1

je booklist

cmp al,2

je exit

jmp invalid

invalid:

;newline

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov ah,1

mov dx,offset a36

int 21h

jmp exit

exit:

call calculate\_total ;calculate total cost

;exit to DOS

MOV AX,4C00H

INT 21H

MAIN ENDP

calculate\_total proc

mov ax,0;

calculate\_loop:

pop cx

add ax,cx

cmp sp,100h

jne calculate\_loop

;display total price message

mov dx,offset total1

mov ah,9

int 21h

;convert total(ax) to ascii & display

call print\_total

calculate\_total endp

;define print\_total procedure to display total price

print\_total PROC

; Convert AX (total price) to ASCII

mov cx, 10 ; Base 10

xor dx, dx ; Clear DX

convert\_loop:

div cx ; Divide AX by 10, remainder in DX

add dl, '0' ; Convert remainder to ASCII

push dx ; Push digit onto stack

xor dx, dx ; Clear DX

cmp ax, 0

jne convert\_loop

print\_digits:

pop dx ; Pop each digit

mov ah, 2

int 21h ; Print the digit

cmp sp, 100H ; Check if stack is empty

jne print\_digits

ret

print\_total ENDP

END MAIN