|  |
| --- |
| THỰC HÀNH VI XỬ LÝ – VI ĐIỀU KHIỂN  GVHD: Bùi Phùng Hữu Đức  Họ và tên sinh viên thực hiện:Nguyễn Hữu Tứ  Mã số sinh viên:19522453 |

BÁO CÁO THỰC HÀNH SỐ 5

1. D;gldkjgldf
2. ẻotoeryt

|  |  |
| --- | --- |
| **Chương trình** | **Ý nghĩa** |
| .MODEL SMALL  .STACK 100H  .DATA  ten DW 10  two DB 2  tb db "Nhap so can chuyen: $ "  tb1 db "So sau khi chuyen la: $"  KQ db 8 dup('0')  temp DW ?  minus DB ?  ;----------------------  MAIN PROC  mov ax,@data  mov ds,ax  mov ah,9  lea dx,tb  int 21h  CALL INPUT  CALL PRINT\_ARRAY  db 13, 10, 0  CALL CONVERT\_BIN  mov ah,9  lea dx,tb1  int 21h  LEA SI, KQ  ADD SI, 7  MOV AH, 0Eh  MOV CX, 8  print\_bin:  MOV AL, [SI]  INT 10h  DEC SI  LOOP print\_bin  MAIN ENDP  ;----------------------  CONVERT\_BIN PROC  PUSH AX  PUSH DI  LEA DI, KQ  MOV AX, CX  MOV AH, 0  next\_step:  DIV CS:two  MOV [DI], AH  ADD [DI], 30h  INC DI  MOV AH, 0  CMP AX, 0  JNE next\_step  do\_reset:  CMP DI, OFFSET KQ + 8  JAE done  MOV [DI], '0'  INC DI  JMP do\_reset  done:  POP DI  POP AX  RET  CONVERT\_BIN ENDP  ;---------------------  INPUT PROC  PUSH DX  PUSH AX  PUSH SI  MOV CX, 0  MOV CS:minus, 0  next:  MOV AH, 00h  INT 16h  MOV AH, 0Eh  INT 10h  CMP AL, 13  JNE not\_cr  JMP stop\_input  not\_cr:  CMP AL, 8  JNE checked  MOV AX, CX  DIV CS:ten  MOV CX, AX  JMP next  checked:  CMP AL, '9'  JBE digit  JMP next  digit:  PUSH AX  MOV AX, CX  MUL CS:ten  MOV CX, AX  POP AX  CMP DX, 0  SUB AL, 30h  MOV AH, 0  MOV DX, CX  ADD CX, AX  JMP next  stop\_input:  CMP CS:minus, 0  JE not\_minus  NEG CX  not\_minus:  POP SI  POP AX  POP DX  RET  INPUT ENDP  ;-----------------------------  PRINT\_ARRAY PROC  MOV CS:temp, SI  POP SI  PUSH AX  next\_char:  MOV AL, CS:[SI]  INC SI  CMP AL, 0  JZ printed  MOV AH, 0Eh  INT 10h  JMP next\_char  printed:  POP AX  PUSH SI  MOV SI, CS:temp  RET  PRINT\_ARRAY ENDP    END MAIN | Khai báo biến ten có giá trị là 10  Khai báo biến two có giá trị là 2  Thông báo chuỗi nhập decimal  Thông báo chuỗi xuất binary  Mảng KQ có 8 phần tử với giá trị là 0  Biến temp không có giá trị ban đầu  Biến minus không có giá trị ban đầu  Hàm chính  Gán địa chỉ của .DATA vào ax  Thực hiện hàm ngắt 9 để in ra 1 chuỗi kí tự thông báo nhập số cần chuyển sang binary  Sau đó, gọi hàm nhập  Xuất giá trị đó ra  Chuỗi cần in  Hàm chuyển sang binary  Thực hiện in chuỗi thông báo số sau khi chuyển là  Gán SI = địa chỉ KQ  SI + 7 (SI = địa chỉ KQ + 7 )  Gán AH = 0Eh (phục vụ ngắt)  Gán CX = 8,số vòng lặp  Hàm này để in ra từng kí tự trong SI từ địa chỉ lớn nhất đến nhỏ nhất.Mục đích in ngược các giá trị được lưu vào stack  Gán AL = [SI] ( = Result + 7)  Ngắt để in kí tự  Giảm địa chỉ SI  Lặp lại  Kết thúc hàm main  //Phần hàm con  Hàm chuyển sang nhị phân  Lưu lại AX vào stack  Lưu lại DI  Gán DI = địa chỉ bộ nhớ KQ  Gán AX = CX  Gán AH = 0  AX = AX/2 số dư lưu vào AH  Gán [DI] = AH  [DI] + 30h để chuyển về ASCII  Tăng DI  Gán AH = 0  So sánh AX với 0  So sánh DI với chỉ số của KQ  Nhảy nếu ZF = 0 và CF = 0  Gán [DI] = ‘0’  Tăng địa chỉ DI  Nhảy tới do\_reset  Hàm nhập kí tự vào  Mục địch nhập vào kí tự.Nếu người dùng nhấn phím enter thì thoát.    Hàm nhận kí tự nhập vào  Push DX(chứa giá trị hex của số nhập)  Push AX(Chứa kí tự cuối của giá trị nhập)  Gán CX = 0  Gán minus = 0  Gán AH = 00h(Phục vụ ngắt)  Ngắt  Gán AH = 0Eh(Phục vụ ngắt)  Ngắt  AL – 13 nếu bằng 0 thì ZF = 1  Nhảy nếu ZF = 0  Nhảy đến stop\_input  AL – 8 nếu = 0 thì ZF = 1  Nhảy nếu ZF = 0  Gán AX = CX  AX = AX/10  Gán CX = AX  Nhảy next  Hàm đê xét nếu chỉ có 1 kí tự  So sánh với 9  Nhảy nếu ZF = 1  Nhảy đến next  Push AX vào stack  Gán AX = CX  AX = AX\*10  Gán CX = AX  Pop AX  AL – 30 để chuyển sang dạng ASCII  Gán AH = 0  Gán DX = CX  CX = CX + AX  Nhảy hàm next  So sánh make với 0  Nhảy nếu điều kiện trên đúng  Đảo dấu CX  Lấy lại giá trị SI  Lấy lại giá trị AX  Lấy lại giá trị DX  Quay lại từ lệnh gọi  Kết thúc hàm  ;--------------------------------  Hàm in ra display  Gán temp = SI  Lấy giá trị vào SI  Đưa giá trị AX vào stack  Gán AL = temp[SI]  Tăng SI (tăng địa chỉ)  So sánh AL với 0  Nhảy nếu ZF = 1  Gán AH = 0Eh để ngắt  Nhảy tới next\_char  Kết thúc hàm  Kết thúc chương trình |

\*Chạy kết quả mô phỏng:

