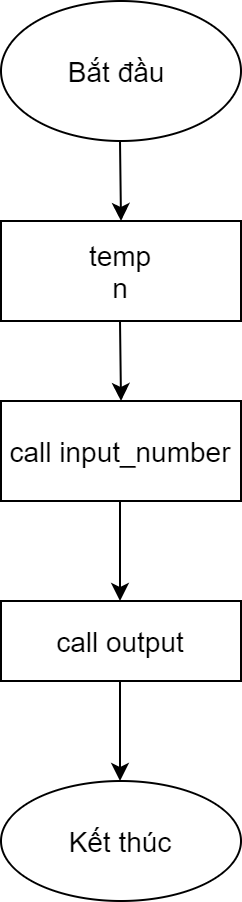
|  |
| --- |
| THỰC HÀNH VI XỬ LÝ – VI ĐIỀU KHIỂN  GVHD: Bùi Phùng Hữu Đức  Họ và tên sinh viên thực hiện: Nguyễn Hữu Tứ  Mã số sinh viên: 19522453 |

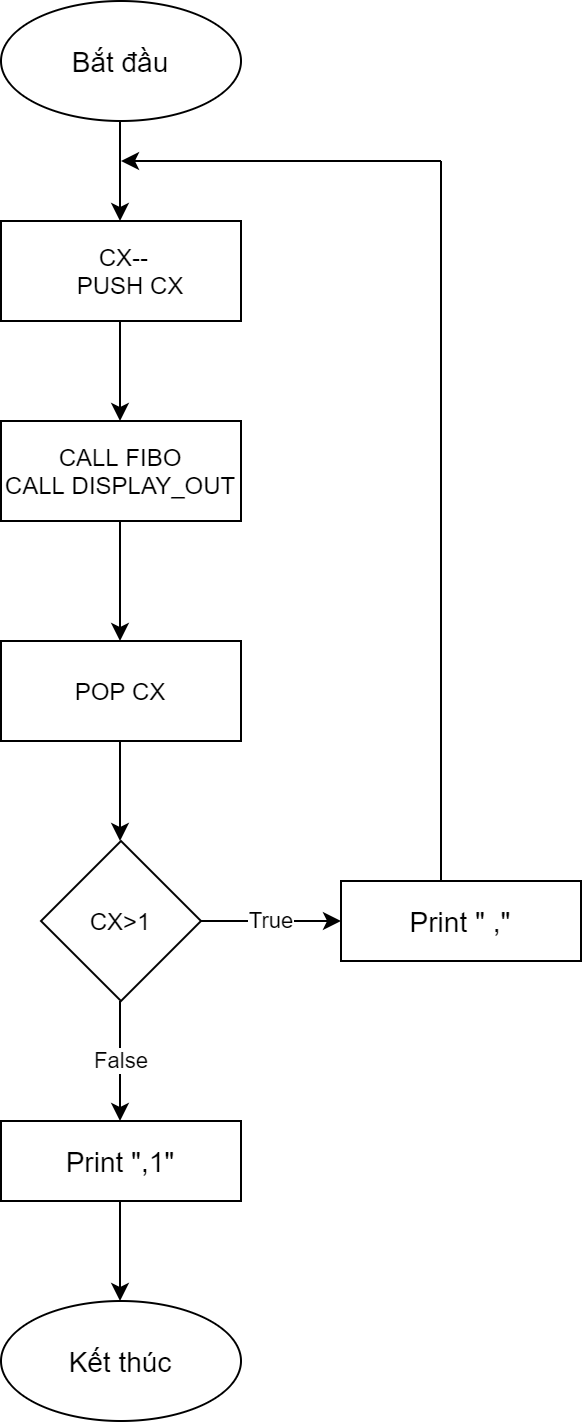
BÁO CÁO THỰC HÀNH SỐ 06

XỬ LÝ IO, TÍNH TOÁN VÀ BỘ NHỚ TRÊN 8086

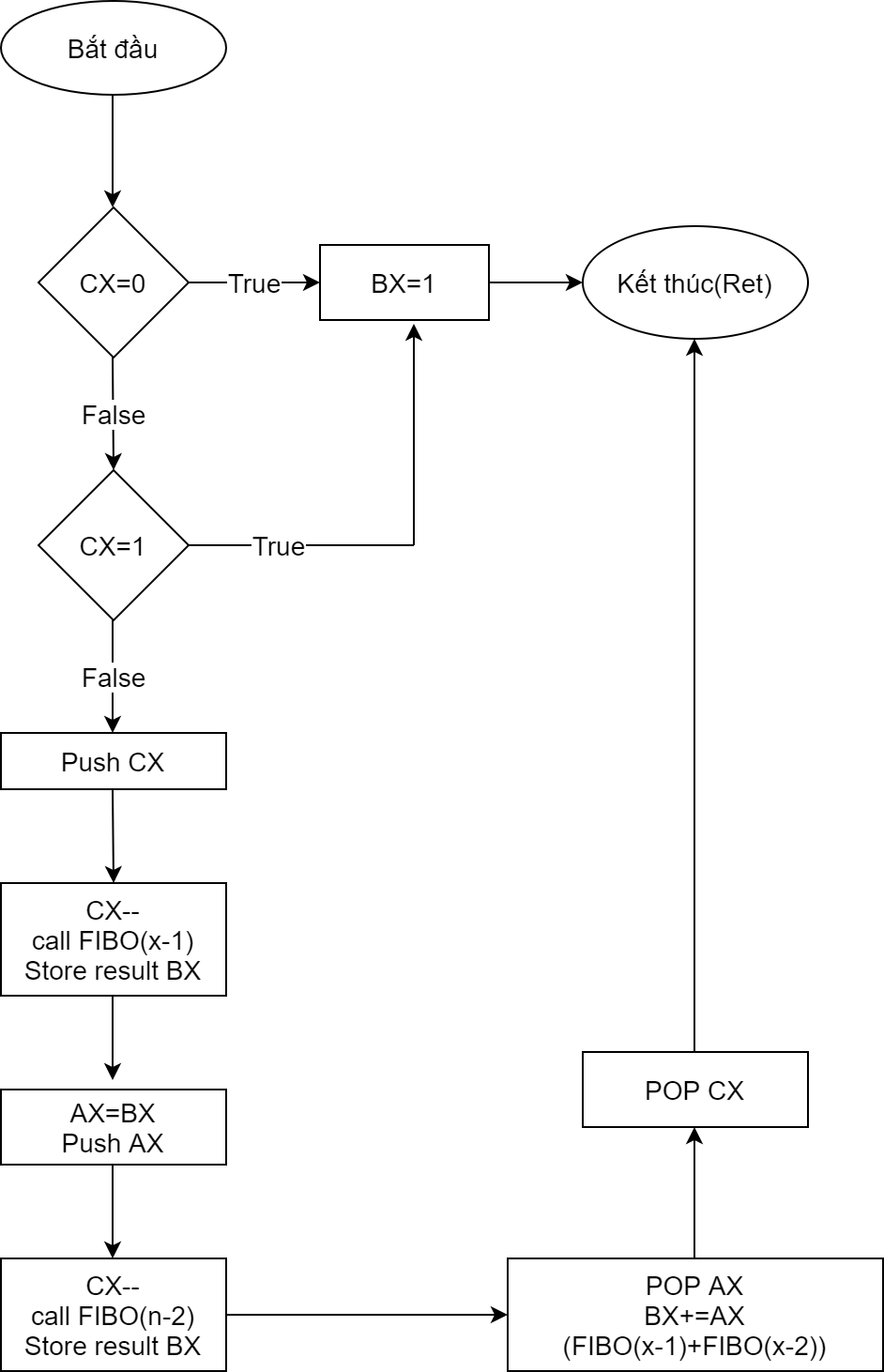
1. Nhập vào một số nguyên N và in ra N số fibonaci đầu tiên :
2. Lưu đồ thuât toán



Hình 1 : Lưu đồ hàm main



Hình 2 :Sơ đồ hàm Output



Hình 3 :Sơ đồ hàm FIBO

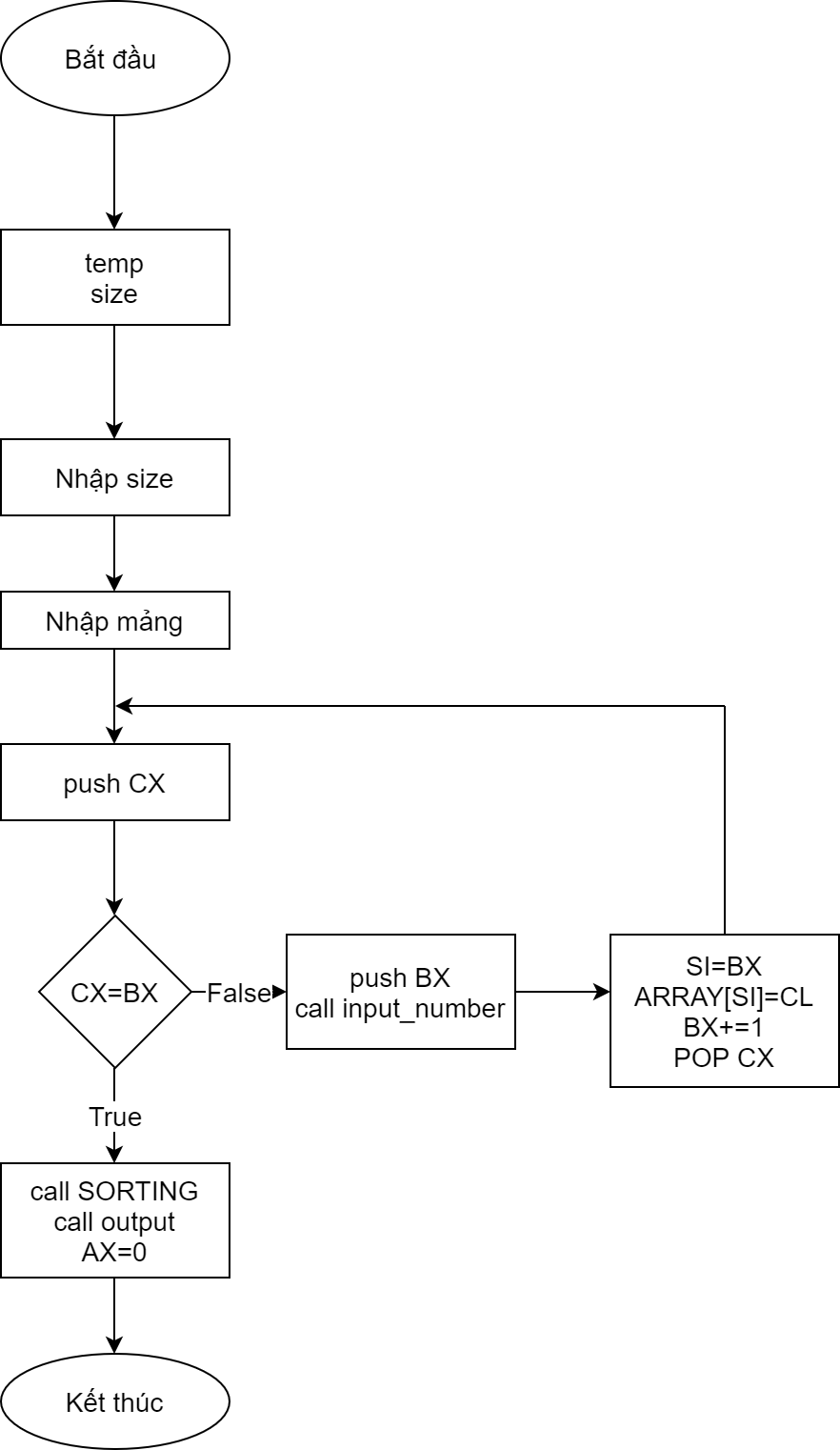
|  |  |
| --- | --- |
| Code | Giải thích |
| .MODEL SMALL  .STACK 100H  .DATA  temp DW ?  tb1 db "Nhap N :$"  under\_line DB 0AH,0DH,"$"  .CODE  MAIN PROC  MOV AX,@DATA  MOV DS,AX  MOV AH,9  LEA DX,tb1  INT 21h  MOV AH,01H  CALL INPUT\_NUMBER  MOV AH,9  LEA DX,under\_line  INT 21h  DB 13,10,0  CALL OUTPUT    MAIN ENDP  ;////////////////////////  DISPLAY\_ PROC  MOV CS:temp, SI  POP SI  PUSH AX  nextchar:  MOV AL, CS:[SI]  INC SI  CMP AL, 0  JZ print  MOV AH, 0Eh  INT 10h  JMP nextchar  print:  POP AX  PUSH SI  MOV SI, CS:temp  RET  DISPLAY\_ ENDP  ;/////////////////////////  INPUT\_NUMBER PROC  MOV BX,0h  MOV CX, 0h  next\_digit:  MOV AH, 00h  INT 16h  MOV AH, 0Eh  INT 10h  CMP AL, 13  JNE next  JMP STOP\_INPUT\_NUMBER  next:  MOV BL,AL  cmp BX,'0'  JB not\_minus  CMP BX, '9'  JA not\_minus  SUB BX,30h  MOV AX, CX  MOV CX, 0AH  MUL CX  MOV CX, AX  ADD CX, BX  JMP next\_digit  not\_minus:  MOV AH,4CH  INT 21h  RET  STOP\_INPUT\_NUMBER:  RET  INPUT\_NUMBER ENDP  ;////////////////////////  FIBO PROC  CMP CX,0H  JE OUT\_FIBO  CMP CX,01H  JE OUT\_FIBO  PUSH CX  DEC CX  CALL FIBO  MOV AX,BX  DEC CX  PUSH AX  CALL FIBO  POP AX  ADD BX,AX  POP CX  RET  OUT\_FIBO:  MOV BX,01H  RET  FIBO ENDP  ;////////////////////////  DISPLAY\_OUT PROC  CMP BX, 0AH  JNB OUTPUT\_NEXT  ADD BX,30h  MOV DX,BX  MOV AH,02H  INT 21H  RET  OUTPUT\_NEXT:  MOV CX, 0H  MOV AX, BX  DISPLAY:  MOV BX, 10D  MOV DX, 00H  DIV BX  ADD DX, 30H  PUSH DX  INC CX  CMP AX, 0H  JE END\_DISPLAY  JMP DISPLAY    END\_DISPLAY:  POP DX  MOV AH, 02H  INT 21H  LOOP END\_DISPLAY  RET  DISPLAY\_OUT ENDP  ;////////////////////////  OUTPUT PROC  DEC CX  PUSH CX  CALL FIBO  CALL DISPLAY\_OUT  POP CX  CMP CX,01H  JA OUT\_SP  JMP PRINT\_1  OUT\_SP:  CALL DISPLAY\_  DB ", ",0  JMP OUTPUT  PRINT\_1:  CALL DISPLAY\_  DB ", 1",0  MOV AH, 4CH  INT 21H    OUTPUT ENDP  ;///////////////////////  END MAIN | Biến temp lưu giá trị tạm với không có giá trị khởi tạo.  Chuỗi tb1 thông báo nhập n  Chuỗi under\_line dùng để xuống dòng  .Phần code  Hàm main  Dùng hàm ngắt để xuất ra chuỗi thông báo nhập n  Hàm nhập số cần tính toán  In ra kí tự xuống dòng để xuất ra chuỗi các số  Ta dùng dùng ngắt  Hàm output xuất giá trị và tính toán  Kết thúc hàm main  Hàm dùng để 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 nextchar  Kết thúc hàm  Hàm INPUT\_NUM BX = 0 CX = 0 AH = 0  Interupt  Interupt So sánh AL = 13 (nhấn Enter) Nhảy nếu không bằng Nhảy về stop\_input\_num  BL = AL So sánh BX >= ‘0’(So sánh ascii) Nhảy nếu điều kiện không đúng So sánh BX <= ‘9’(So sánh ascii) Nhảy nếu điều kiện trên không đúng  BX - 30h (Chuyển về kí tự ascii) AX = CX CX = 10 AX\*CX (= AX \* 10) CX = AX CX + BX Nhảy  Interupt terminal  Trở về từ lệnh call  Hàm tính FIBO So sánh CX = 0 Bằng thì nhảy OUT\_FIBO So sánh CX = 1 Bằng thì nhảy OUT\_FIBO Lưu CX CX -= 1 Gọi FIBO (Fibo(CX - 1)) AX = BX (Hàm trên kết quả lưu BX) CX -= 1 Lưu AX Gọi FIBO(Fibo(CX - 2)) Lấy AX BX = AX+BX= ( Fibo(CX-1) + Fibo(CX-2) ) Trở về từ hàm gọi  BX = 1  Trở về từ hàm gọi  Kết thúc hàm  Hàm DISPLAY\_OUT So sánh BX > 10 Nhảy nếu điều kiện trên sai BX + 30h DX = BX Interupt In kí tự Trở về từ lệnh gọi  CX = 0  AX = BX  BX = 10 DX = 0 AX = AX/BX DX + 30 (DX = AX%BX) Lưu DX CX + 1 So sánh AX = 0 Nhảy nếu bằng Nhảy DISPLAY Lấy DX  Interupt In kí tự  LOOP  Trở về từ lệnh CALL  Kết thúc hàm  Giảm CX Lưu CX Gọi hàm FIBO(tính số fibo) Gọi DISPLAY\_OUT  So sánh CX > 1 Không đúng thì nhảy OUT\_SP (in dấu ,) Nhảy PRINT\_1  Hàm in ra màn hình  Nhảy về OUTPUT(lặp lại với giá trị CX giảm dần) Hàm in ra màn hinh In kí tự 1 (Chuỗi fibo đã in xong)  Ngắt Terminal  Kết thúc hàm  Kết thúc chương trình |

-Kết quả mô phỏng:

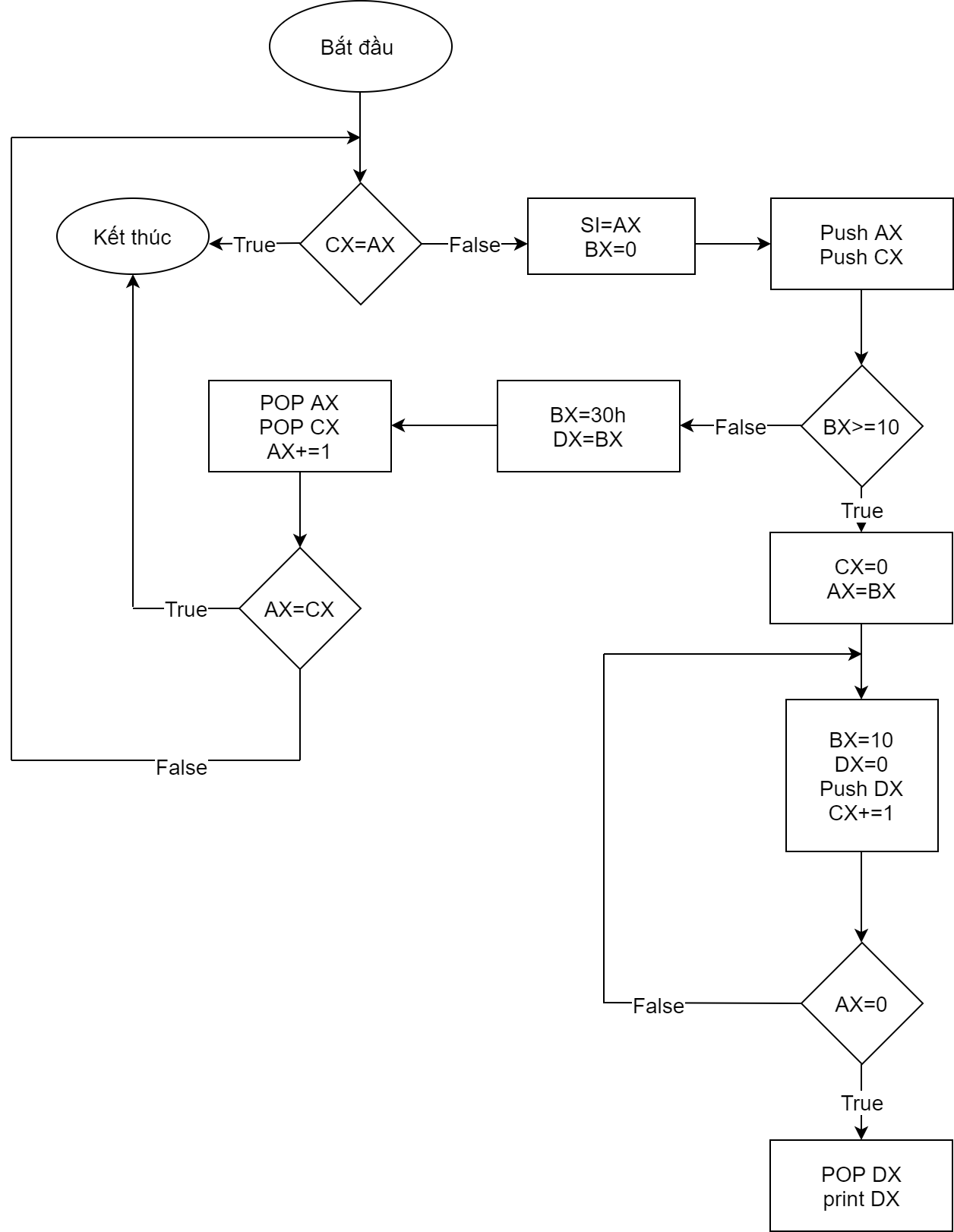


Hình 4 : Kết quả mô phỏng

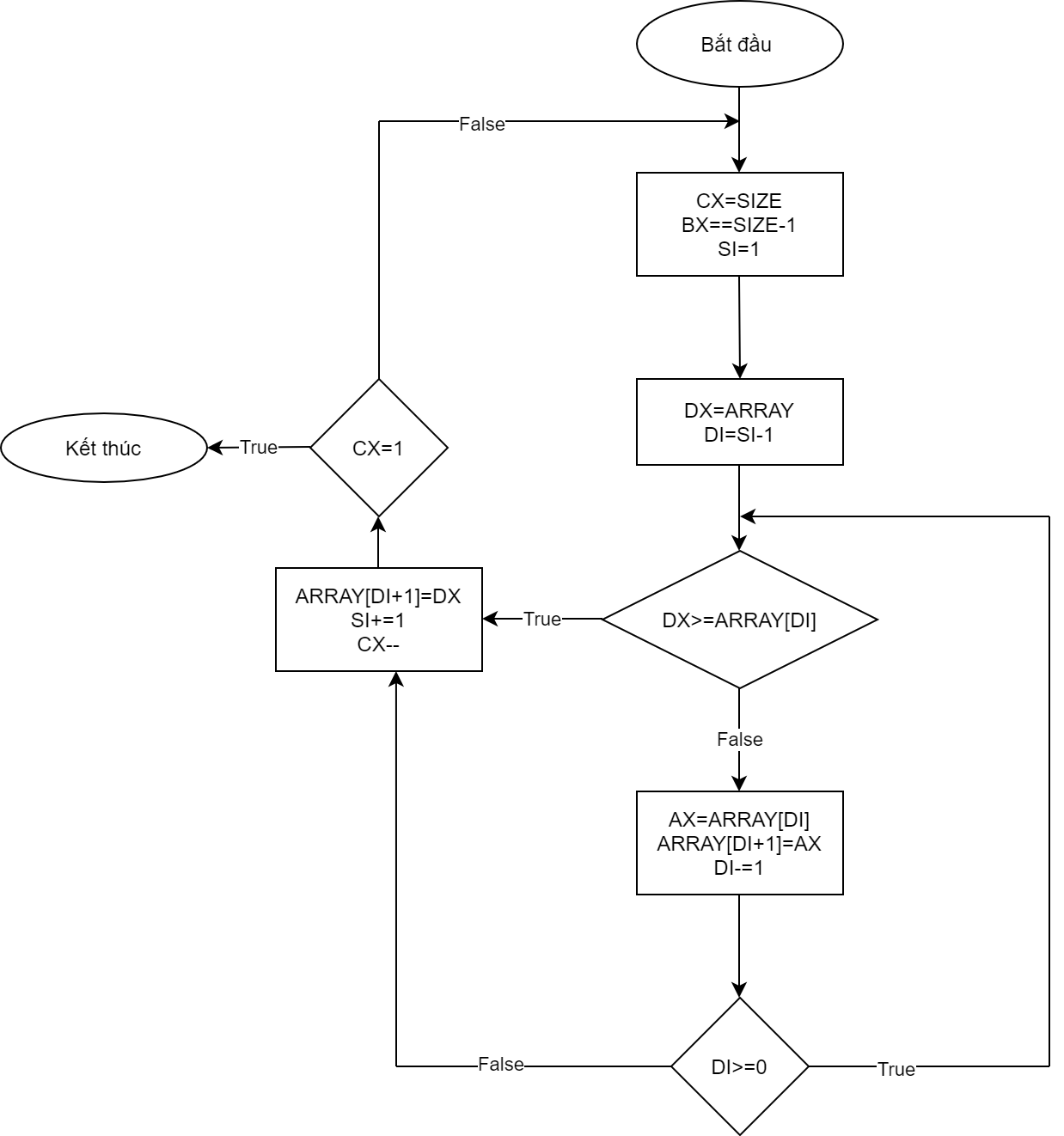
1. Nhập mảng N(N<15), sắp xếp theo thứ tự tăng dần và in ra màn hình console:
2. Sơ đồ giải thuật :



Hình 5 : Sơ đồ hàm main



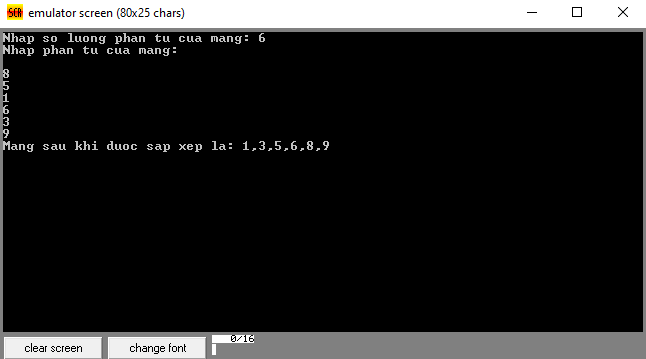
Hình 6 : Sơ đồ hàm output



Hình 7 : Sơ đồ hàm Sort

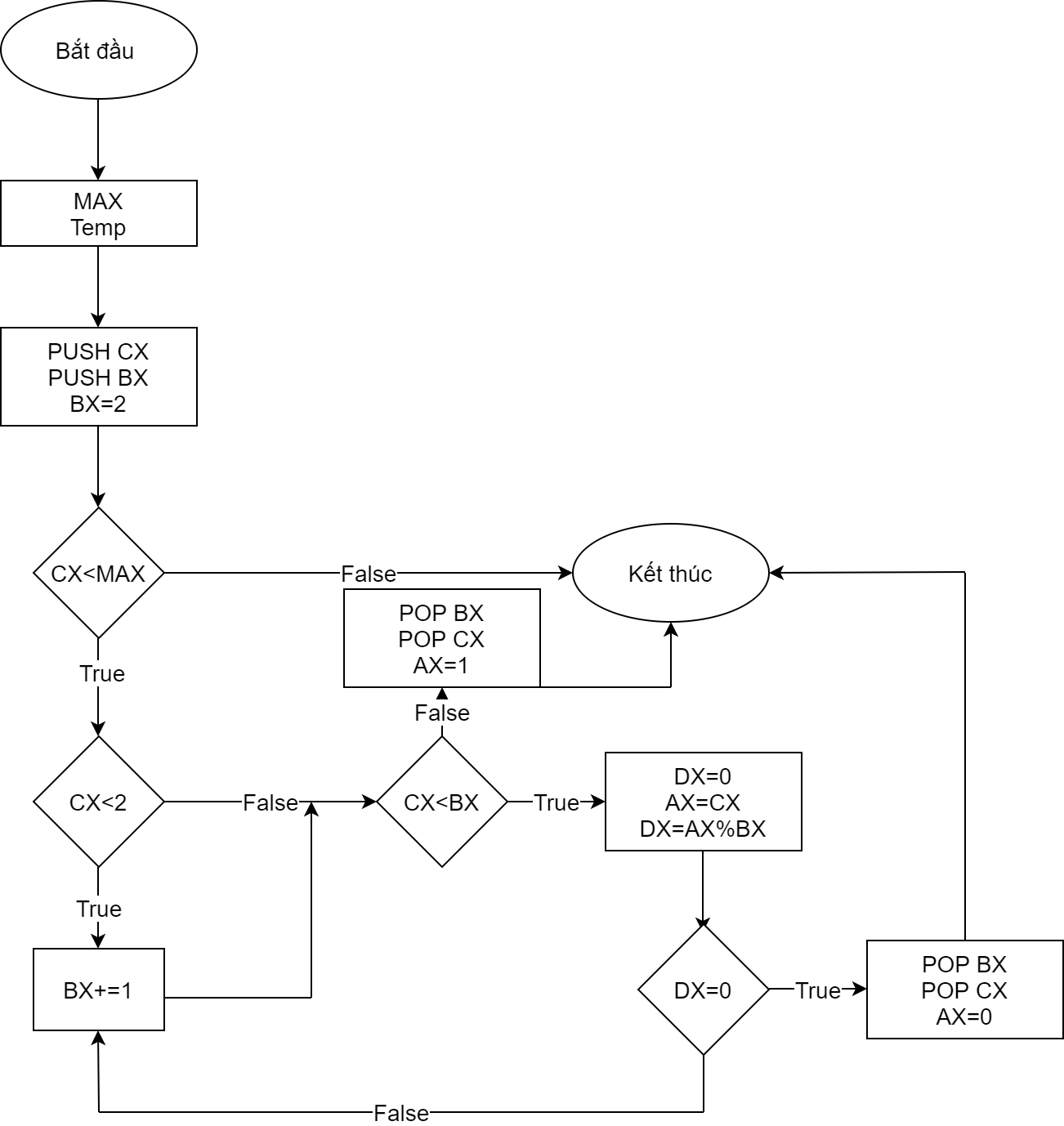
|  |  |
| --- | --- |
| Code | Ý nghĩa |
| .MODEL SMALL  org 100h  .STACK 100H  .DATA  ARRAY DB 15 DUP(?)  tb1 db "Nhap so luong phan tu cua mang: $"  tb2 db "Nhap phan tu cua mang: $"  tb3 db "Mang sau khi duoc sap xep la: $"  tb4 db ", $"  under\_line DB 0AH,0DH,"$"    SIZE DW ?  temp DW ?    .CODE  MAIN PROC    MOV AX,@DATA  MOV DS,AX  MOV AL , ARRAY  OUTPUT\_TB1:    MOV AH,9  LEA DX,tb1  INT 21h  MOV AH,01H  CALL INPUT\_NUMBER  MOV SIZE,BX    OUTPUT\_TB2:  MOV BX ,00H    MOV AH,9  LEA DX,under\_line  INT 21h    MOV AH,9  LEA DX,tb2  INT 21h      MOV AH,9  LEA DX,under\_line  INT 21h    INPUT\_ARRAY:    MOV AH,9  LEA DX,under\_line  INT 21h    PUSH CX  CMP BX, CX  JE OUTPUT\_SORT  PUSH BX  CALL INPUT\_NUMBER      STOP\_INPUT:  POP BX  MOV SI, BX  MOV ARRAY[SI], CL  INC BX  POP CX  JMP INPUT\_ARRAY    OUTPUT\_SORT:  CALL SORTING    MOV AH,9  LEA DX,tb3  INT 21h  MOV AX , 00H      CALL OUTPUT        MAIN ENDP  ;////////////////////////////    INPUT\_NUMBER PROC  MOV BX,0h  MOV CX, 0h  next\_digit:  MOV AH, 00h  INT 16h  MOV AH, 0Eh  INT 10h  CMP AL, 13  JNE next  JMP STOPINPUT\_NUMBER  next:  MOV BL,AL  cmp BX,'0'  JB not\_minus  CMP BX, '9'  JA not\_minus  SUB BX,30h  MOV AX, CX  MOV CX, 0AH  MUL CX  MOV CX, AX  ADD CX, BX  JMP next\_digit  not\_minus:  MOV AH,4CH  INT 21h  RET  STOPINPUT\_NUMBER:  RET  INPUT\_NUMBER ENDP  ;////////////////////////////  SORTING PROC  PUSHA  MOV CX,CS:SIZE  MOV BX,CX  SUB BX,1  MOV SI,1  FOR\_LOOP:  MOV DL,ARRAY[SI]  MOV DI,SI  SUB DI,1  WHILE\_LOOP:  CMP DL,ARRAY[DI]  JGE EXIT\_WHILE\_LOOP  MOV AL,ARRAY[DI]  MOV ARRAY[DI+1],AL  SUB DI,1  CMP DI,0  JGE WHILE\_LOOP  EXIT\_WHILE\_LOOP:  MOV ARRAY[DI+1],DL  ADD SI,1  DEC CX  CMP CX,1  JNE FOR\_LOOP  SORT\_DONE:  POPA  RET 4    SORTING ENDP  ;///////////////////////////  OUTPUT\_C PROC  MOV CS:temp, SI  POP SI  PUSH AX  nextchar:  MOV AL, CS:[SI]  INC SI  CMP AL, 0  JZ print  MOV AH, 0Eh  INT 10h  JMP nextchar  print:  POP AX  PUSH SI  MOV SI, CS:temp  RET  ;/////////////////////  OUTPUT PROC    CMP AX, CX  JE not\_minus    MOV SI , AX  MOV BX, 00H  MOV BL , ARRAY[SI]  PUSH CX  PUSH AX    CMP BX, 0AH  JNB OUTPUT\_NEXT  ADD BX ,30h  MOV DX, BX  MOV AH, 02H  INT 21H  JMP OUT\_1  OUTPUT\_NEXT:  MOV CX, 00H  MOV AX, BX  DISPLAY:  MOV BX, 10D  MOV DX, 00H  DIV BX  ADD DX, 30H  PUSH DX  INC CX  CMP AX, 0  JE END\_DISPLAY  JMP DISPLAY    END\_DISPLAY:  POP DX  MOV AH, 02H  INT 21H  LOOP END\_DISPLAY    OUT\_1:  POP AX  POP CX  INC AX  CMP AX, CX  JE not\_minus  CALL OUTPUT\_C  DB ",",0      JMP OUTPUT  RET  OUTPUT ENDP  ;/////////////////////    END MAIN | Tạo mảng  Các thông báo báo nhập xuất  Xuống dòng  Kích thước mảng  Biến tạm temp  Phần .code  Hàm chính  AX = @DATA DS = AX AL = ARRAY  Hàm ngắt in ra màn hình In ra màn hình  AH = 1 Hàm nhập từ bàn phím SIZE = BX  BX = 0  Hàm ngắt in ra màn hình Xuống dòng Hàm in ra màn hình  Lưu vào CX So sánh BX = CX Bằng thì nhảy  Gọi hàm nhập (nhập các phần tử)  SI = BX ARRAY[SI]= CL Tăng BX lên 1 Nhảy đến INPUT\_ARRAY để nhập tiếp phần tử tiếp theo  Gọi hàm sort  Hàm in ra màn hình  AX = 0  Hàm xuất  Kết thúc hàm chính  Hàm nhập INPUT BX = 0 CX = 0  AH = 0 Interupt AH = 0EH Interupt So sánh AL = 13 (Enter) Nhảy nếu không bằng Nhảy BL = AL So sánh BX >= ‘0’(so sánh ascii)  Nhảy nếu không đúng  So sánh BX < =‘9’(so sánh ascii) Nhảy nếu không đúng BX – 30H(Chuyển về mã ascii) AX = CX CX = 10 AX = AX\*10 CX = AX CX = BX (Cộng số còn lại) Nhảy  Interupt  Quay về từ lệnh gọi  Kết thúc hàm  Hàm sort Lưu register CX = SIZE BX = SIZE BX = SIZE – 1 SI = 1 DL = ARRAY[SI] DI = SI  DI = DI – 1  So sánh DL >= ARRAY[DI] Đúng thì nhảy AL = ARRAY[DI] ARRAY[DI+1] = AL DI = DI – 1 So sánh DI > = 0 Đúng thì nhảy  ARRAY[DI+1] = DL SI = SI + 1 Giảm CX So sánh CX = 1 Sai thì nhảy  Kết thúc hàm sort  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 nextchar  Kết thúc hàm  So sánh AX = CX Đúng thì nhảy  SI = AX BX = 0 BL = ARRAY[SI]  So sánh BX >= 10 Không đúng thì nhảy BX + 30h (chuyển sang ascii) DX = BX Interupt để in kí tự  Nhảy đến OUT\_1  CX = 0 AX = BX  BX = 10 DX = 0 AX = AX / 10 DX + 30h  Tăng CX thêm 1  So sánh AX = 0  Bằng thì nhảy Nhảy đến DISPLAY  Tăng AX thêm 1 So sánh AX = CX Nhảy nếu bằng In ra màn hình  Kết thúc hàm  Kết thúc chương trình |

-Kết quả mô phỏng :



Hình 8 : Mô phỏng kết quả

1. Nhập vào số nguyên N (N<150),in ra các số nguyên tố nhỏ hơn N vào file.txt( mỗi số cách nhau một khoảng trắng)
2. Sơ đồ giải thuật:



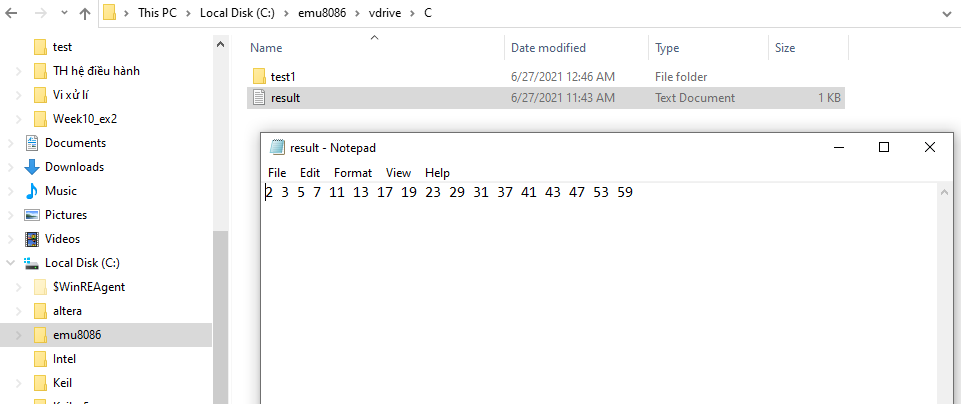
Hình 9 : Sơ đồ giải thuật bài 3

|  |  |
| --- | --- |
| Code | Ý nghĩa |
| .MODEL SMALL  include 'emu8086.inc'  org 100h  .DATA  tb1 db "Nhap so N: $"  tb2 db "Cac so nguyen to: $"  under\_line DB 0AH,0DH,"$"  space db " $"  outfile DB "c:\result.txt",0  MAX dw ?  temp db ?    .CODE  MAIN PROC    MOV AX,@DATA  MOV DS,AX    DISPLAY\_TB1:    MOV AH,9  LEA DX,tb1  INT 21h  CALL INPUT\_NUMBER      MOV AH,9  LEA DX,under\_line  INT 21h  DISPLAY\_TB2:  MOV AH,9  LEA DX,tb2  INT 21h      PUSH CX  mov ah, 3ch  mov cx, 0  mov dx, offset outfile  int 21h  jc EXIT  mov MAX, ax  POP CX      MOV BX, 00H    LOOP\_PRINT:  PUSH CX  MOV CX , BX    CALL PRIME\_FUNCTION  POP CX    CMP AX, 01H    JE PRINT  JMP NEXT  MAIN ENDP  ;///////////////////////////  INPUT\_NUMBER PROC  MOV CX, 0H  MOV BX, 0H  READ:  MOV AH,01H  INT 21H    CMP AL,0DH    JZ EXIT1    MOV BL,AL    cmp BX,'0'  JB EXIT  cmp BX,'9'  JA EXIT    SUB BX, 30H    MOV AX, CX  MOV CX, 0AH    MUL CX  MOV CX, AX  ADD CX, BX      JMP READ  EXIT1:  RET    INPUT\_NUMBER ENDP    ;//////////////////////////  PRIME\_FUNCTION PROC    PUSH CX  PUSH BX  CMP CX , 02H  JB NOT\_PRIME      MOV BX, 02H  LOOP\_PRIME\_NUMBER:    CMP BX, CX  JB CONTINUE\_PRIME\_FUNCTION  JMP PRIME    CONTINUE\_PRIME\_FUNCTION:  MOV DX, 00H  MOV AX, CX  DIV BX  CMP DX, 00H  JE NOT\_PRIME    INC BX  JMP LOOP\_PRIME\_NUMBER:      PRIME:  POP BX  POP CX  MOV AX, 01H  RET        NOT\_PRIME:  POP BX  POP CX  MOV AX, 00H  RET    PRIME\_FUNCTION ENDP  ;/////////////////////////////  DISPLAY\_NUMBER PROC  CMP BX, 0AH  JB LESS\_THAN\_10  MOV CX, 00H  MOV AX, BX  CONTINUE\_DISPLAY:  MOV BX, 10D  MOV DX, 00H  DIV BX  ADD DX, 30H  PUSH DX  INC CX  CMP AX, 0  JE DISPLAY\_END  JMP CONTINUE\_DISPLAY            LESS\_THAN\_10:  ADD BX , 30h    CALL WRITE\_FILE    MOV DX, BX  MOV AH, 02H  INT 21H  RET    DISPLAY\_END:  POP DX  MOV BX, DX    CALL WRITE\_FILE    MOV AH, 02H  INT 21H      LOOP DISPLAY\_END    RET  DISPLAY\_NUMBER ENDP    ;///////////////////////////  WRITE\_FILE PROC    PUSH BX  PUSH cx  PUSH DX  mov ah, 40h  MOV temp, BL  mov bx, MAX  mov dx, offset temp  mov cx, 01h  int 21h  POP DX  POP cx  POP BX    RET    WRITE\_FILE ENDP  ;///////////////////////////  PRINT:        PUSH CX  PUSH BX  CALL DISPLAY\_NUMBER    mov ah, 40h  MOV temp, 32d  mov bx, MAX  mov dx, offset temp  mov cx, 01h  int 21h  POP BX  POP CX      MOV AH,9  LEA DX,space  INT 21h  NEXT:  INC BX    LOOP LOOP\_PRINT      EXIT:  mov ah, 4ch  int 21h  RET  ;/////////////////////////////////  END MAIN | Phần .Data  Khai báo nhập xuất  Phần xuống dòng  Phần khoảng trắng  File để ghi kết quả vào  Biến MAX  Biến tạm Temp  Phần Code  Hàm chính  Hàm thông báo nhập số  Dùng hàm ngắt để in ra chuỗi cần thông báo  Gọi hàm input\_number để nhập số từ bàn phím  Dùng hàm ngắt để hiển thị xuống dòng  Dùng hàm ngắt để thông báo in các số nguyên tố  Lưu CX AH = 3CH CX = 0 DX, offset outfile  So sánh AX với 0, đúng thì thoát CT  MAX=ax  Lấy giá trị CX  BX=0  Vòng lặp in ra các số nguyên tố  Lưu giá trị của CX  CX=BX  Hàm sử lí nguyên tố  Lấy giá trị của CX  So sánh AX với 01  Nếu AX=1 thì nhảy tới hàm print  Nếu không thì nhảy tới hàm NEXT  Kết thúc hàm chính  Hàm nhập giá trị  CX=0  BX=0  AH=01  Interrupt  So sánh AL với 0Dh  Nhảy nếu =0  BL = AL  So sánh BX >= ‘0’(so sánh ascii) Nhảy nếu không đúng So sánh BX < =‘9’(so sánh ascii)  Nhảy nếu không đúng  BX – 30H(Chuyển về mã ascii)  AX = CX CX = 10  AX = AX\*10 CX = AX CX = BX (Cộng số còn lại)  Nhảy  Kết thúc hàm nhập  Hàm xử lý số nguyên tố  Lưu CX Lưu BX So sánh CX với 2  Nhảy nếu CX nhỏ hơn 2  BX=2  So sánh BX với CX  Nhảy nếu BX<CX  Nhảy không thì nhảy tới hàm PRIME  DX = 0 AX = CX AX = AX/BX So sánh DX = 0 (DX = AX%BX) Bằng thì nhảy NOT\_PRIME  BX++  Lấy lại BX Lấy CX AX = 01 Trở về  Lấy lại BX Lấy CX AX = 0 Trở về  Kết thúc hàm  So sánh BX với 10  Nhảy nếu BX<10  CX=0  AX=BX  BX=10  DX=0  AX = AX/BX DX + 30 (DX = AX%BX) Lưu DX  CX++  So sánh AX với 0  Nhảy nếu bằng Nhảy CONTINUE\_DISPLAY  Hàm nhở hơn 10  BX+=30h  Gọi hàm ghi vào file  DX=BX  Interrupt in kí tự  Trở về hàm chính  Lấy DX  BX=DX  Gọi hàm ghi file  Interrupt in kí tự  Kết thúc hàm  Hàm ghi vào file  Lưu giá trị của BX  Lưu giá trị của CX  Lưu giá trị của DX  Ah=40h  Temp=BL  Bx =MAX  Thực hiện ghi  Interrupt  Lấy giá trị của BX  Lấy giá trị của CX  Lấy giá trị của DX  Trở về hàm chính  Kết thúc hàm ghi  Lưu giá trị CX  Lưu giá trị BX  Gọi hàm DISPLAY\_NUMBER  Ah=40h  Temp=32  Bx=MAX  Thực hiện ghi  Interrupt  Lấy giá trị của BX  Lấy giá trị của CX  In ra kí tự khoảng trắng  BX++  Hàm thoát chương trình  Kết thúc chương trình |

-Kết quả mô phỏng:



Hình 10 : Mô phỏng kết quả bài 3



Hình 11 : Ghi vào file Result.txt