## THỰC HÀNH VI XỬ LÝ – VI ĐIỀU KHIỂN

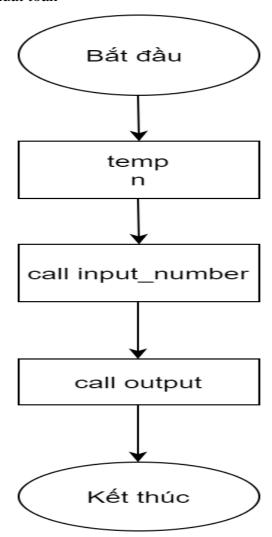
GVHD: Bùi Phùng Hữu Đức

Họ và tên sinh viên thực hiện: Nguyễn Hữu Tứ

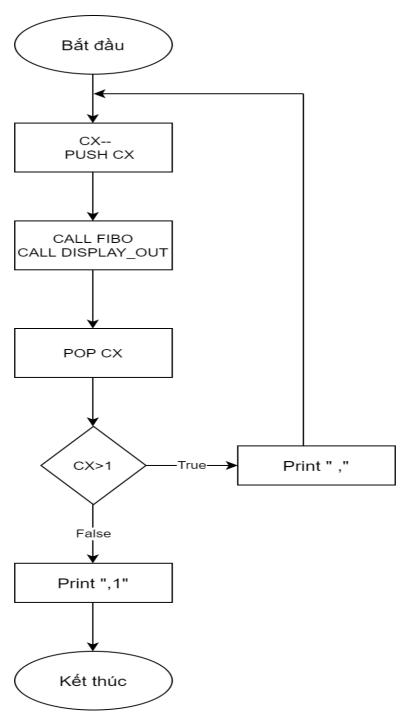
Mã số sinh viên: 19522453

## BÁO CÁO THỰC HÀNH SỐ 06 XỬ LÝ IO, TÍNH TOÁN VÀ BỘ NHỚ TRÊN 8086

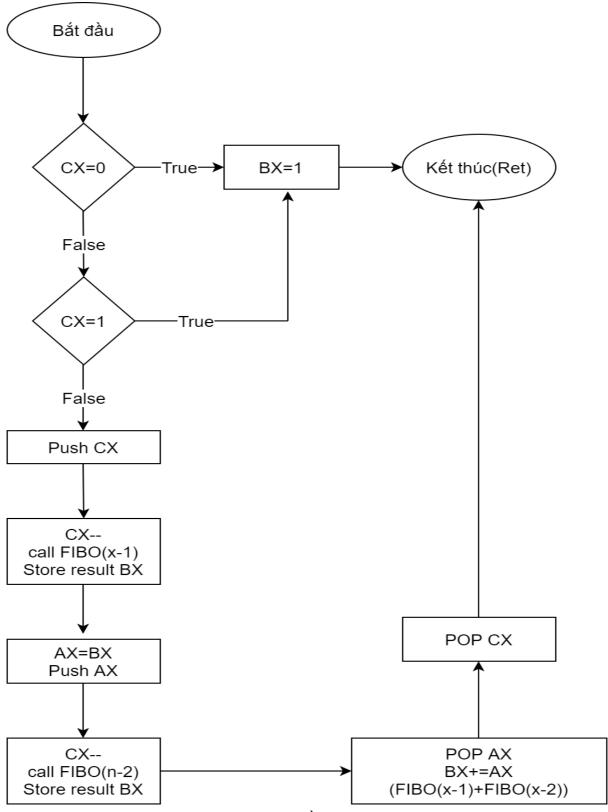
- I. Nhập vào một số nguyên N và in ra N số fibonaci đầu tiên:
  - 1. Lưu đồ thuật toán



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	Diân tama lama aid tui tama sadi labân a ad
.DATA	Biến temp lưu giá trị tạm với không có
temp DW ?	giá trị khởi tạo.
tb1 db "Nhap N :\$"	Chuỗi tb1 thông báo nhập n
under_line DB 0AH,0DH,"\$"	Chuỗi under_line dùng để xuống dòng
.CODE	.Phần code
MAIN PROC	Hàm main
MOV AX,@DATA	
MOV DS,AX	
MOV AH,9	Dùng hàm ngắt để xuất ra chuỗi thông
LEA DX,tb1	báo nhập n
INT 21h	oao map n
MOV AH,01H	
CALL INPUT_NUMBER	Hàm nhập số cần tính toán
MOV AH,9	In ra kí tự xuống dòng để xuất ra chuỗi
LEA DX,under_line	các số
INT 21h	Ta dùng dùng ngắt
DB 13,10,0	
CALL OUTPUT	Hàm output xuất giá trị và tính toán
MAIN ENDP	Kết thúc hàm main
;///////	
DISPLAY_PROC	Hàm dùng để display
MOV CS:temp, SI	Gán temp = SI
POP SI	Lấy giá trị vào SI
PUSH AX	Dua giá trị AX vào stack
nextchar:	Dua gia ti AA vao stack
	Cán AI tomo [CI]
MOV AL, CS:[SI]	Gán AL = temp[SI] $T = SL(t) = \frac{1}{2}$
INC SI	Tăng SI (tăng địa chỉ)
CMP AL, 0	So sánh AL với 0
JZ print	Nhảy nếu ZF = 1
MOV AH, 0Eh	Gán AH = 0Eh để ngắt
INT 10h	
JMP nextchar	Nhảy tới nextchar
print:	
POP AX	
PUSH SI	
MOV SI, CS:temp	
RET	
DISPLAY_ENDP	Kết thúc hàm
;//////////////////////////////////////	
INPUT_NUMBER PROC	Hàm INPUT_NUM
MOV BX,0h	BX = 0
MOV CX, 0h	CX = 0
next_digit:	AH = 0
MOV AH, 00h	
1710 Y 7111, UUII	<u>l</u>

INT 16h	
MOV AH, 0Eh	Interupt
INT 10h	1
CMP AL, 13	Interupt
JNE next	So sánh AL = 13 (nhấn Enter)
JMP STOP_INPUT_NUMBER	Nhảy nếu không bằng
next:	Nhảy về stop_input_num
MOV BL,AL	Timey ve stop_mput_num
cmp BX,'0'	BL = AL
JB not_minus	So sánh BX $\geq$ = '0'(So sánh ascii)
CMP BX, '9'	Nhảy nếu điều kiện không đúng
JA not_minus	So sánh BX <= '9'(So sánh ascii)
SUB BX,30h	Nhảy nếu điều kiện trên không đúng
MOV AX, CX	BX - 30h (Chuyển về kí tự ascii)
	AX = CX
MOV CX, 0AH MUL CX	AX = CX CX = 10
MOV CX, AX	$AX^*CX (= AX * 10)$
ADD CX, BX	CX = AX
JMP next_digit	CX + BX
not_minus:	Nhảy
MOV AH,4CH	T
INT 21h	Interupt terminal
RET	T *
STOP_INPUT_NUMBER:	Trở về từ lệnh call
RET	
INPUT_NUMBER ENDP	
.//////////////////////////////////////	
;/////////////////////////////////////	Hàm tính EIDO
FIBO PROC	Hàm tính FIBO
CMP CX,0H	So sánh $CX = 0$
JE OUT_FIBO	Bằng thì nhảy OUT_FIBO
CMP CX,01H	So sánh $CX = 1$
JE OUT_FIBO	Bằng thì nhảy OUT_FIBO
PUSH CX	Lru CX
DEC CX	CX -= 1
CALL FIBO	Gọi FIBO (Fibo(CX - 1))
MOV AX,BX	AX = BX (Hàm trên kết quả lưu $BX$ )
DEC CX	CX -= 1
PUSH AX	Lru AX
CALL FIBO	Gọi FIBO(Fibo(CX - 2))
POP AX	Lấy AX
ADD BX,AX	BX = AX + BX = (Fibo(CX-1) +
POP CX	Fibo(CX-2))
RET	Trở về từ hàm gọi
OUT_FIBO:	
MOV BX,01H	BX = 1

RET	Tuảy việ từ hòm coi
	Trở về từ hàm gọi Kết thúc hàm
FIBO ENDP	Ket thuc nam
;//////////////////////////////////////	
DISPLAY_OUT PROC	Hàm DISPLAY_OUT
CMP BX, 0AH	So sánh BX > 10
JNB OUTPUT_NEXT	Nhảy nếu điều kiện trên sai
ADD BX,30h	BX + 30h
MOV DX,BX	DX = BX
MOV AH,02H	Interupt In kí tự
INT 21H	Trở về từ lệnh gọi
RET	Tro ve tu remi ger
OUTPUT_NEXT:	
MOV CX, 0H	CX = 0
MOV AX, BX	AX = BX
DISPLAY:	
MOV BX, 10D	BX = 10
MOV DX, 00H	DX = 0
DIV BX	AX = AX/BX
ADD DX, 30H	DX + 30 (DX = AX%BX)
PUSH DX	Lưu DX
INC CX	CX + 1
CMP AX, 0H	So sánh $AX = 0$
JE END_DISPLAY	Nhảy nếu bằng
JMP DISPLAY	Nhảy DISPLAY
	Lấy DX
END_DISPLAY:	
POP DX	Interupt In kí tự
MOV AH, 02H	
INT 21H	
LOOP END_DISPLAY	LOOP
RET	Trở về từ lệnh CALL
DISPLAY_OUT ENDP	Kết thúc hàm
;//////////////////////////////////////	
OUTPUT PROC	Giảm CX
	Lưu CX
DEC CX	Gọi hàm FIBO(tính số fibo)
PUSH CX	Goi DISPLAY_OUT
CALL FIBO	
CALL DISPLAY_OUT	So sánh CX > 1
POP CX	Không đúng thì nhảy OUT_SP (in dấu,)
CMP CX,01H	Nhảy PRINT_1
JA OUT_SP	-
JMP PRINT_1	Hàm in ra màn hình
OUT_SP:	
CALL DISPLAY_	

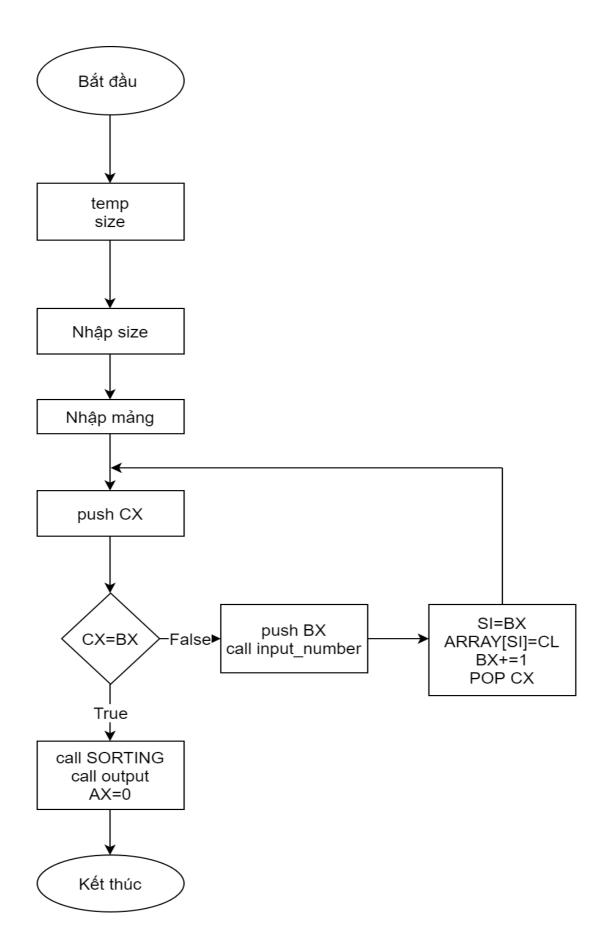
DB ", ",0 Nhảy về OUTPUT(lặp lại với giá trị CX giảm dần) JMP OUTPUT Hàm in ra màn hinh PRINT\_1: In kí tự 1 (Chuỗi fibo đã in xong) CALL DISPLAY\_ DB ", 1",0 Ngắt Terminal MOV AH, 4CH **INT** 21H Kết thúc hàm **OUTPUT ENDP** Kết thúc chương trình **END MAIN** 

## -Kết quả mô phỏng:

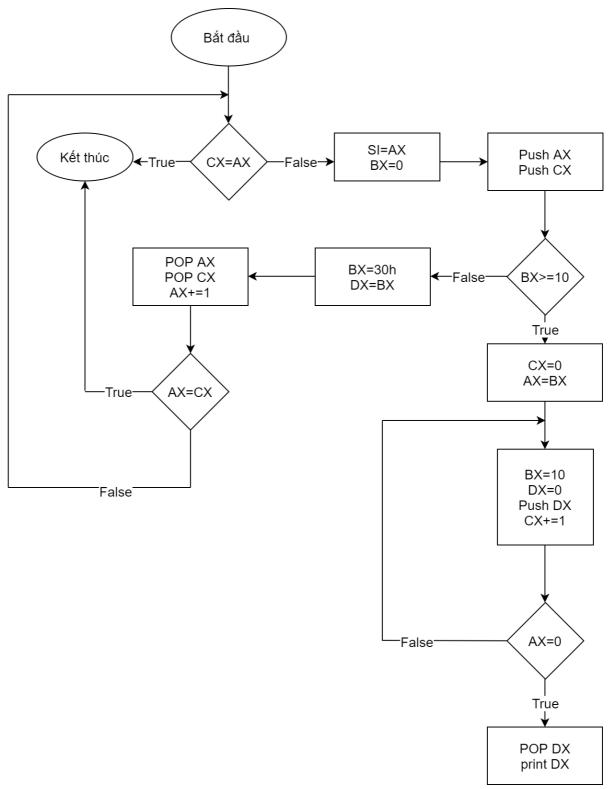


Hình 4 : Kết quả mô phỏng

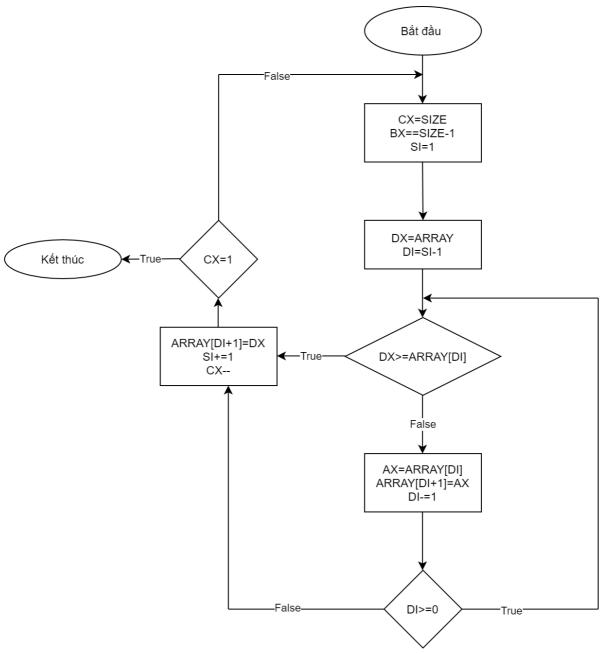
- II. Nhập mảng N(N<15), sắp xếp theo thứ tự tăng dần và in ra màn hình console:
  - 1. Sơ đồ giải thuật:



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(?)	Tạo mảng
tb1 db "Nhap so luong phan tu cua	Các thông báo báo nhập xuất
mang: \$"	
tb2 db "Nhap phan tu cua mang: \$"	
tb3 db "Mang sau khi duoc sap xep la:	
\$"	

	T
tb4 db ", \$"	_
under_line DB 0AH,0DH,"\$"	Xuống dòng
	Kích thước mảng
SIZE DW ?	Biến tạm temp
temp DW ?	
temp bw:	Phần .code
CODE	
.CODE	Hàm chính
MAIN PROC	
	AX = @DATA
MOV AX,@DATA	DS = AX
MOV DS,AX	AL = ARRAY
MOV AL , ARRAY	
OUTPUT_TB1:	
	Hàm ngắt in ra màn hình
MOV AH,9	In ra màn hình
LEA DX,tb1	
*	
INT 21h	
	AH = 1
MOV AH,01H	Hàm nhập từ bàn phím
CALL INPUT_NUMBER	SIZE = BX
MOV SIZE,BX	
OUTPUT_TB2:	
MOV BX ,00H	BX = 0
7	
MOV AH,9	Hàm ngắt in ra màn hình
LEA DX,under_line	Xuống dòng
INT 21h	Hàm in ra màn hình
1111 2111	
MOV AH,9	
· ·	
LEA DX,tb2	
INT 21h	
MOV AV S	
MOV AH,9	
LEA DX,under_line	
INT 21h	
INPUT_ARRAY:	
MOV AH,9	
LEA DX,under_line	
INT 21h	
1111 2111	Lưu vào CX
DUCHCY	So sánh BX = CX
PUSH CX	Bằng thì nhảy
CMP BX, CX	Dang un may

JE OUTPUT_SORT	
PUSH BX	
CALL INPUT_NUMBER	Gọi hàm nhập (nhập các phần tử)
STOP_INPUT:	
POP BX	SI = BX
MOV SI, BX	ARRAY[SI]= CL
MOV ARRAY[SI], CL	Tăng BX lên 1
INC BX	Nhảy đến INPUT ARRAY để nhập
POP CX	1 . · · · . —
	tiếp phần tử tiếp theo
JMP INPUT_ARRAY	
OUTPUT SORT:	Gọi hàm sort
CALL SORTING	- ; <del></del>
	Hàm in ra màn hình
MOV AH,9	
·	
LEA DX,tb3 INT 21h	
	AV
MOV AX , 00H	AX = 0
CALL OUTPUT	Hàm xuất
CALLOUITUI	
MAIN ENDP	Kết thúc hàm chính
;//////////////////////////////////////	
····	
INPUT_NUMBER PROC	Hàm nhập INPUT
MOV BX,0h	BX = 0
MOV CX, 0h	CX = 0
next_digit:	
	AH = 0
MOV AH, 00h	Interupt
INT 16h	AH = 0EH
MOV AH, 0Eh	
INT 10h	Interupt
CMP AL, 13	So sánh $AL = 13$ (Enter)
JNE next	Nhảy nếu không bằng
JMP STOPINPUT_NUMBER	Nhảy
next:	BL = AL
MOV BL,AL	So sánh BX >= '0'(so sánh ascii)
cmp BX,'0'	
JB not_minus	Nhảy nếu không đúng
CMP BX, '9'	So sánh BX < = '9'(so sánh ascii)
JA not_minus	Nhảy nếu không đúng
JI II II II III II II II II II II II II	

	2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
SUB BX,30h	BX – 30H(Chuyển về mã ascii)
MOV AX, CX	AX = CX
MOV CX, 0AH	CX = 10
MUL CX	AX = AX*10
MOV CX, AX	CX = AX
ADD CX, BX	CX = BX (Cộng số còn lại)
JMP next_digit	Nhảy
not_minus:	
MOV AH,4CH	Interupt
INT 21h	•
RET	Quay về từ lệnh gọi
STOPINPUT_NUMBER:	
RET	
INPUT NUMBER ENDP	Kết thúc hàm
;///////	2200 02000 22002
SORTING PROC	Hàm sort
PUSHA	Luu register
MOV CX,CS:SIZE	CX = SIZE
MOV BX,CX	BX = SIZE
SUB BX,1	BX = SIZE - 1
MOV SI,1	SI = 1
FOR_LOOP:	DL = ARRAY[SI]
<u> </u>	DI = ARRA I [SI] DI = SI
MOV DLSI	D1 – S1
MOV DI,SI	DI DI 1
SUB DI,1	DI = DI - 1
WHILE_LOOP:	
CMP DL,ARRAY[DI]	So sánh DL $\geq$ = ARRAY[DI]
JGE EXIT_WHILE_LOOP	Đúng thì nhảy
MOV AL,ARRAY[DI]	AL = ARRAY[DI]
MOV ARRAY[DI+1],AL	ARRAY[DI+1] = AL
SUB DI,1	DI = DI - 1
CMP DI,0	So sánh DI $> = 0$
JGE WHILE_LOOP	Đúng thì nhảy
EXIT_WHILE_LOOP:	
MOV ARRAY[DI+1],DL	ARRAY[DI+1] = DL
ADD SI,1	SI = SI + 1
DEC CX	Giảm CX
CMP CX,1	So sánh CX = 1
JNE FOR_LOOP	Sai thì nhảy
SORT_DONE:	
POPA	
RET 4	
SORTING ENDP	Kết thúc hàm sort
;//////////////////////////////////////	
OUTPUT_C PROC	

MOV CS:temp, SI	Hàm in ra display
POP SI	Gán temp = SI
PUSH AX	Lấy giá trị vào SI
nextchar:	Đưa giá trị AX vào stack
MOV AL, CS:[SI]	<i>S</i> .
INC SI	Gán AL = temp[SI]
CMP AL, 0	Tăng SI (tăng địa chỉ)
JZ print	So sánh AL với 0
MOV AH, 0Eh	Nhảy nếu ZF = 1
INT 10h	Tillay lieu Zi
JMP nextchar	Gán AH = 0Eh để ngắt
print:	Nhảy tới nextchar
POP AX	Tillay tor nextenar
PUSH SI	
MOV SI, CS:temp	
RET	
;/////////////////////////////////////	Kết thúc hàm
OUTPUT PROC	Ket that ham
OUTFORTROC	
CMP AX, CX	
JE not_minus	So sánh $AX = CX$
JE not_minus	
MOVSLAV	Đúng thì nhảy
MOV SI, AX	SI = AX
MOV BX, 00H	
MOV BL , ARRAY[SI]	BX = 0
PUSH CX	BL = ARRAY[SI]
PUSH AX	
CMD DV OAH	
CMP BX, 0AH	Co cánh DV > = 10
JNB OUTPUT_NEXT	So sánh BX >= $10$
ADD BX ,30h	Không đúng thì nhảy
MOV DX, BX	BX + 30h (chuyển sang ascii)
MOV AH, 02H	$DX = BX$ Let a most $\frac{1}{2}$ in $\frac{1}{2}$ in $\frac{1}{2}$ in $\frac{1}{2}$
INT 21H	Interupt để in kí tự
JMP OUT_1	Nhảy đến OUT_1
OUTPUT_NEXT:	CV 0
MOV CX, 00H	CX = 0
MOV AX, BX	AX = BX
DISPLAY:	DV 10
MOV BX, 10D	BX = 10
MOV DX, 00H	DX = 0
DIV BX	AX = AX / 10
ADD DX, 30H	DX + 30h
PUSH DX	
INC CX	Tăng CX thêm 1
CMP AX, 0	So sánh $AX = 0$

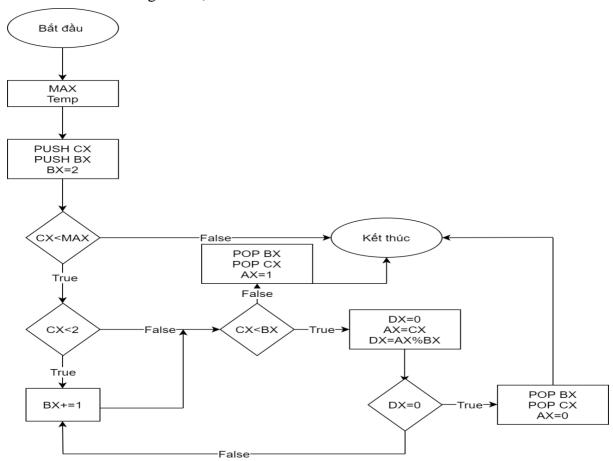
	_ <b>'</b>
JE END_DISPLAY	Bằng thì nhảy
JMP DISPLAY	Nhảy đến DISPLAY
END_DISPLAY:	
POP DX	
MOV AH, 02H	
INT 21H	
LOOP END_DISPLAY	
EOOI END_DISTENT	
OUT_1:	
POP AX	
POP CX	T~ A 37 (1 ^ 1
INC AX	Tăng AX thêm 1
CMP AX, CX	So sánh $AX = CX$
JE not_minus	Nhảy nếu bằng
CALL OUTPUT_C	In ra màn hình
DB ",",0	
JMP OUTPUT	
RET	
OUTPUT ENDP	Kết thúc hàm
;//////////////////////////////////////	The time ham
3,((()))	
END MAIN	Kết thúc chương trình
ENDWAIN	Ket that chaong trim

-Kết quả mô phỏng:



Hình 8: Mô phỏng kết quả

- III. 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)</p>
  - 1. Sơ đồ giải thuật:



Hình 9 : Sơ đồ giải thuật bài 3

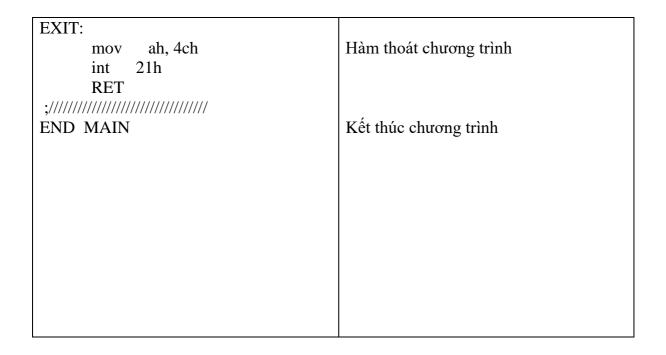
Code	Ý nghĩa
.MODEL SMALL	5
include 'emu8086.inc'	
org 100h	
.DATA	Phần .Data
tb1 db "Nhap so N: \$"	Khai báo nhập xuất
tb2 db "Cac so nguyen to: \$"	
under_line DB 0AH,0DH,"\$"	Phần xuống dòng
space db " \$"	Phần khoảng trắng
outfile DB "c:\result.txt",0	File để ghi kết quả vào
MAX dw?	Biến MAX
temp db?	Biển tạm Temp
.CODE	Phần Code
MAIN PROC	Hàm chính
MOV AX,@DATA	
MOV DS,AX	
DISPLAY_TB1:	Hàm thông báo nhập số
MOV AH,9	Dùng hàm ngắt để in ra chuỗi cần thông
LEA DX,tb1	báo
INT 21h	
CALL INPUT_NUMBER	Gọi hàm input_number để nhập số từ bàn phím
MOV AH,9	Dùng hàm ngắt để hiển thị xuống dòng
LEA DX,under_line	Bung main ngar ar men mi maong aong
INT 21h	
DISPLAY_TB2:	Dùng hàm ngắt để thông báo in các số
MOV AH,9	nguyên tố
LEA DX,tb2	
INT 21h	
PUSH CX	Luu CX
mov ah, 3ch	AH = 3CH
mov cx, 0	CX = 0
mov dx, offset outfile	DX, offset outfile
int 21h	
jc EXIT	So sánh AX với 0, đúng thì thoát CT
mov MAX, ax	MAX=ax
POP CX	Lấy giá trị CX

MOV BX, 00H	BX=0
1.13 + B11, 0011	
LOOP_PRINT:	Vòng lặp in ra các số nguyên tố
PUSH CX	Lưu giá trị của CX
MOV CX , BX	CX=BX
,	
CALL PRIME_FUNCTION	Hàm sử lí nguyên tố
POP CX	Lấy giá trị của CX
	<i>y 8</i>
CMP AX, 01H	So sánh AX với 01
,	
JE PRINT	Nếu AX=1 thì nhảy tới hàm print
JMP NEXT	Nếu không thì nhảy tới hàm NEXT
MAIN ENDP	Kết thúc hàm chính
;///////	
INPUT_NUMBER PROC	Hàm nhập giá trị
MOV CX, 0H	CX=0
MOV BX, 0H	BX=0
READ:	
MOV AH,01H	AH=01
INT 21H	Interrupt
	1
CMP AL,0DH	So sánh AL với 0Dh
JZ EXIT1	Nhảy nếu =0
MOV BL,AL	BL = AL
cmp BX,'0'	So sánh BX >= '0'(so sánh ascii)
JB EXIT	Nhảy nếu không đúng
cmp BX,'9'	So sánh BX <='9'(so sánh ascii)
JA EXIT	Nhảy nếu không đúng
SUB BX, 30H	BX – 30H(Chuyển về mã ascii)
MOV AX, CX	AX = CX
MOV CX, 0AH	CX = 10
MUL CX	AX = AX*10
MOV CX, AX	CX = AX
ADD CX, BX	CX = BX (Cộng số còn lại)
JMP READ	Nhảy
EXIT1:	
RET	

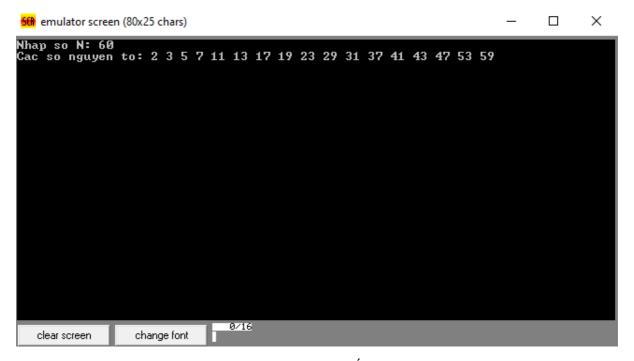
Kết thúc hàm nhập INPUT\_NUMBER ENDP Hàm xử lý số nguyên tố PRIME FUNCTION PROC **PUSH CX** Lưu CX **PUSH BX** Lưu BX CMP CX, 02H So sánh CX với 2 JB NOT PRIME Nhảy nếu CX nhỏ hơn 2 MOV BX, 02H BX=2LOOP\_PRIME\_NUMBER: CMP BX, CX So sánh BX với CX JB CONTINUE\_PRIME\_FUNCTION Nhảy nếu BX<CX Nhảy không thì nhảy tới hàm PRIME JMP PRIME CONTINUE\_PRIME\_FUNCTION: DX = 0MOV DX, 00H AX = CXMOV AX, CX DIV BX AX = AX/BXSo sánh DX = 0 (DX = AX%BX) CMP DX, 00H JE NOT\_PRIME Bằng thì nhảy NOT\_PRIME INC BX BX++JMP LOOP\_PRIME\_NUMBER: PRIME: Lấy lại BX POP BX Lấy CX POP CX AX = 01MOV AX, 01H Trở về **RET** NOT\_PRIME: Lấy lại BX POP BX Lấy CX POP CX AX = 0MOV AX, 00H Trở về **RET** Kết thúc hàm PRIME\_FUNCTION ENDP

DISPLAY\_NUMBER PROC CMP BX, 0AH So sánh BX với 10 JB LESS\_THAN\_10 Nhảy nếu BX<10 MOV CX, 00H CX=0MOV AX, BX AX=BXCONTINUE\_DISPLAY: MOV BX, 10D BX=10MOV DX, 00H DX=0AX = AX/BXDIV BX ADD DX, 30H DX + 30 (DX = AX%BX)**PUSH DX** Lưu DX CX++INC CX CMP AX, 0 So sánh AX với 0 JE DISPLAY\_END Nhảy nếu bằng JMP CONTINUE\_DISPLAY Nhảy CONTINUE\_DISPLAY LESS\_THAN\_10: Hàm nhở hơn 10 ADD BX, 30h BX+=30hCALL WRITE\_FILE Gọi hàm ghi vào file MOV DX, BX DX=BXInterrupt in kí tự MOV AH, 02H INT 21H Trở về hàm chính **RET** DISPLAY\_END: Lấy DX POP DX BX=DX MOV BX, DX Gọi hàm ghi file CALL WRITE\_FILE MOV AH, 02H Interrupt in kí tự INT 21H LOOP DISPLAY\_END **RET** Kết thúc hàm DISPLAY\_NUMBER ENDP 

WRITE_FILE PROC	Hàm ghi vào file
PUSH BX	Lưu giá trị của BX
PUSH cx	Lưu giá trị của CX
PUSH DX	Lưu giá trị của DX
mov ah, 40h	Ah=40h
MOV temp, BL	Temp=BL
mov bx, MAX	Bx =MAX
mov dx, offset temp	Thực hiện ghi
mov cx, 01h	Interrupt
int 21h	
POP DX	Lấy giá trị của BX
POP cx	Lấy giá trị của CX
POP BX	Lấy giá trị của DX
RET	Trở về hàm chính
WRITE_FILE ENDP	Kết thúc hàm ghi
;//////////////////////////////////////	
PRINT:	
PUSH CX	Lưu giá trị CX
PUSH BX	Lưu giá trị BX
	_
PUSH BX	Lưu giá trị BX
PUSH BX CALL DISPLAY_NUMBER	Lưu giá trị BX Gọi hàm DISPLAY_NUMBER
PUSH BX CALL DISPLAY_NUMBER mov ah, 40h	Lưu giá trị BX Gọi hàm DISPLAY_NUMBER  Ah=40h
PUSH BX CALL DISPLAY_NUMBER  mov ah, 40h MOV temp, 32d	Lưu giá trị BX Gọi hàm DISPLAY_NUMBER Ah=40h Temp=32
PUSH BX CALL DISPLAY_NUMBER  mov ah, 40h MOV temp, 32d mov bx, MAX	Lưu giá trị BX Gọi hàm DISPLAY_NUMBER  Ah=40h Temp=32 Bx=MAX
PUSH BX CALL DISPLAY_NUMBER  mov ah, 40h MOV temp, 32d mov bx, MAX mov dx, offset temp	Lưu giá trị BX Gọi hàm DISPLAY_NUMBER  Ah=40h Temp=32 Bx=MAX Thực hiện ghi Interrupt
PUSH BX CALL DISPLAY_NUMBER  mov ah, 40h MOV temp, 32d mov bx, MAX mov dx, offset temp mov cx, 01h	Lưu giá trị BX Gọi hàm DISPLAY_NUMBER  Ah=40h Temp=32 Bx=MAX Thực hiện ghi Interrupt  Lấy giá trị của BX
PUSH BX CALL DISPLAY_NUMBER  mov ah, 40h MOV temp, 32d mov bx, MAX mov dx, offset temp mov cx, 01h int 21h	Lưu giá trị BX Gọi hàm DISPLAY_NUMBER  Ah=40h Temp=32 Bx=MAX Thực hiện ghi Interrupt
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	Lưu giá trị BX Gọi hàm DISPLAY_NUMBER  Ah=40h Temp=32 Bx=MAX Thực hiện ghi Interrupt  Lấy giá trị của BX
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	Lưu giá trị BX Gọi hàm DISPLAY_NUMBER  Ah=40h Temp=32 Bx=MAX Thực hiện ghi Interrupt  Lấy giá trị của BX
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	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
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	Lưu giá trị BX Gọi hàm DISPLAY_NUMBER  Ah=40h Temp=32 Bx=MAX Thực hiện ghi Interrupt  Lấy giá trị của BX
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	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
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:	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
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	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
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	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
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:	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



## -Kết quả mô phỏng:



Hình 10 : Mô phỏng kết quả bài 3