GOVT. POLYTECHNIC
AMRAVATI

Vision

Provide skilled professionals in Computer Engineering to contribute towards the advancement of technology useful for society and industrial environment.

Mission

M1. Impart need based and value based education by providing exposure of latest tools and technologies in the area of computer engineering to satisfy the stakeholders.

M2. Upgrade and maintain facilities for quality technical education with continuous effort for excellence in Computer Engineering.

M3. Train students with Computer Engineering knowledge to apply it in the general disciplines of design, deployment of software and integration of existing technologies for E-governance and for benefit of society.

M4. Provide a learning ambience to enhance innovations, problem solving skills, leadership qualities, team spirit and ethical responsibilities.

M5. Provide an academic environment and consultancy services to the industry and society in the area of Computer Engineering.

MICRO-PROJECT REPORT

ON

Generate Fibonacci series.

In Partial fulfilment of Diploma in Computer Engineering

In the subject of

MICROPROCESSORS (CM3409)

By

Mr. Ayush Bulbule (19CM007)

Submitted To

Government Polytechnic, Amravati

(An Autonomous Institute of Govt. of Maharashtra)

Under the guidance of

Mrs. S. R. Patil Mam

Lecturer in Microprocessors
Department of Computer Science & Engg.
Government Polytechnic, Amravati
(2020-2021)

Government Polytechnic, Amravati.

(An Autonomous Institute of Govt. of Maharashtra)

Department of Computer Science & Engg.

Certificate

This is to certify that **Mr. Ayush Bulbule (19CM007)** of Third Semester Diploma in Computer Engineering has satisfactorily completed the micro project entitled "**Generate Fibonacci Series**." in **(CM3409) Microprocessors** the academic year 2020-21 as prescribed in curriculum.

Place: Amravati

Date: 05 /02 /2021

Mrs. S. R. Patil Mam

Lecturer in **Microprocessors**

Annexure-I

Title of Micro-Project

Generate Fibonacci Series

1.0 Brief Introduction

The project entitled “Generate Fibonacci Series” is a micro project build in Assembly programming Language. This project is Assembly Language Program to generate Fibonacci Series.

.

2.0 Aim of the Micro-Project

This Micro-Project aims at: Generate Fibonacci Series using Assembly Program

1. Write the program development steps.
2. Explain the standard format to write program.
3. Describe different assembly directives

3.0 Action Plan (Sequence and time required for major activities for 8 weeks)

S. N.	Details of activity	Planned Start date	Planned Finish date	I. Code &Name of Team Members
1	Gathering Information	04-05-2020	08-05-2021	Bhagyashree Tekade (19CM003)
2	Making report and file	14-05-2021	17-05-2021	Pratham Gaur (19CM020)
3	Preparing the Code of project in Assembly Lang.	16-05-2021	20-05-2021	Ayush Bulbule (19CM007)
4	Planning proposal submission	12-05-2021	13-05-2021	Akanksha Shewatkar (19CM057)
5	Gathering content	8-05-2020	11-05-2020	Malhar Joshi (19CM033)

4.0 Resources Required (major resources such as raw material, some machining facility, software etc.)

S.N.	Name of Resource/material	Specifications	Remarks
1	Computer System (System with basic configuration)	__	
2.	8086 Assemblr		
3	TASM 1.4		
4	DOS BOX 0.74		
5	M S Word		
6	Windows 10 OS		

5.0 Names of Team Members with Identity Codes :

i. Bhagyashree Tekade (19CM003)
ii. Ayush Bulbule (19CM007)
iii. Pratham Gaur (19CM020)
iv. Malhar Joshi (19CM033)
v. Akanksha Shewatkar (19CM057)

Guideline for Assessment of Micro-Project

Evaluation as per suggested Rubric for Assessment of Micro-Project

Assessment Parameter	Characteristic to be assessed	Average (1 mark)	Good (1.5 mark)	Excellent (2 mark)
Process Assessment (06)	Relevance of the courses & proposals			
	Literature survey/market survey/information collection			
	Analysis of data & completion of the target as per proposal/			
Product Assessment (04)	Report Preparation/Quality of Prototype/model			

Annexure-II

Title of Micro Project

Generate Fibonacci Series

1.0 Brief Introduction

The project entitled "Generate Fibonacci Series" is a mini project build in Assembly programming Language. This project is Assembly Language Program to generate Fibonacci Series.

2.0 Aim of the Micro-Project

This Micro-Project aims at : 'Generating Fibonacci Series'

1. Write the program development steps.
2. Explain the standard format to write program.
3. Describe different assembly directives

3.0 Course Outcomes Integrated

1. Learn concepts of Assembly Programming

4.0 Actual Procedure Followed

1) Bhagyashree Tekade: Gathered information about the Assembly Programming

2) Ayush Bulbule: Prepared the Code for the program in Assembly

3) Pratham Gaur: Prepared word file related the project with synopsis also.

4) Malhar Joshi: Tested the system and gathered other related info.

5) Akanksha Shewatkar: Planned about and managed submission.

5.0 Actual Resources used (Mention the actual resources used)

S.N.	Name of Resource/material	Specifications	Remarks
1	Computer System (System with basic configuration)	__	
2	TASM	***version*** 1.4	
3	DOS BOX	Version – 0.74	

4	M S Word	2019	
5	Windows 10	--	

6.0 Output of the Micro-Project

Output of this Micro-Project is attached to this file.

7.0 Skill Developed / Learning outcomes of this Micro-Project

1. Learned Assembly Programming.
2. Logic Building in Assembly Language Programming.

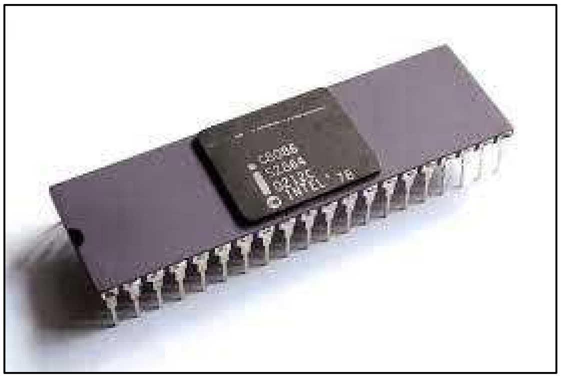
8.0 Assessment by Faculty as per Rubrics

Process Assessment (06)	Product Assessment (04)	Total Marks (10)	Signature of Faculty

****

Microprocessor

A **microprocessor** is an electronic component that is used by a computer to do its work. It is a central processing unit on a single integrated circuit chip containing millions of very small components including transistors, resistors, and diodes that work together. Some microprocessors in the 20th century required several chips. Microprocessors help to do everything from controlling elevators to searching the Web. Everything a computer does is described by instructions of computer programs, and microprocessors carry out these instructions many millions of times a second.



Microprocessor operations

Like other central processing units, microprocessors use three steps commonly called Fetch, Decode, and Execute. In the Fetch step, an instruction is copied from the compute memory into the microprocessor. In the Decode step, the microprocessor figures out what operation the instruction is meant to do. In the Execute step, this operation is performed. Different computers can have different instruction sets.

Brief History

- 1823 Baron Jöns Jacob Berzelius discovers silicon (Si), which today is the basic component of electronics.
- 1903 Nikola Tesla patents electrical logic circuits called "gates" or "switches".
- 1947 John Bardeen, Walter Brattain, and William Shockley invent the first transistor at the Bell Laboratories on December 23, 1947.
- 1956 John Bardeen, Walter Brattain, and William Shockley are awarded the Nobel Prize in physics for their work on the transistor.
- 1958 The first integrated circuits are developed by Robert Noyce and Jack Kilby.

- 1960 IBM develops the first automatic mass-production facility for transistors in New York.
- 1971 Intel 4004, first commercially available microprocessor.

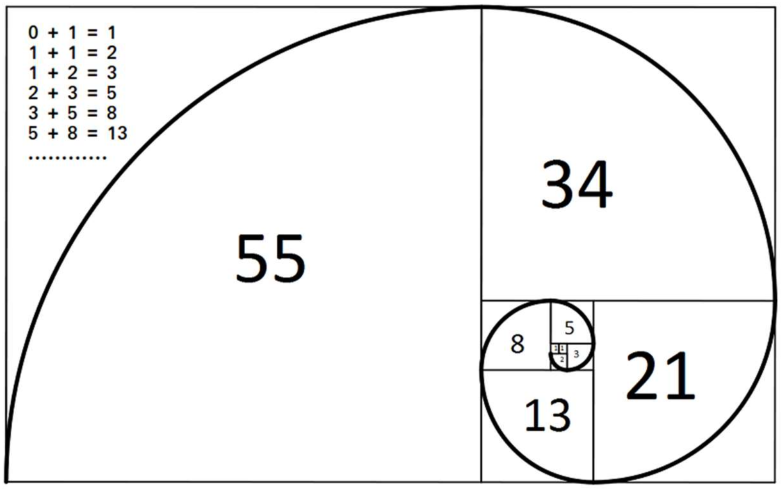
Generate Fibonacci Series

About Project

The Fibonacci numbers are the numbers in the following integer sequence.
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ……..

In mathematical terms, the sequence Fn of Fibonacci numbers is defined by the recurrence relation

Working Diagram:



Instruction Set used in Program :

MOV – Used to copy the byte or word from the provided source to the provided destination.

OR − Used to multiply each bit in a byte/word with the corresponding bit in another byte/word.

INT − Used to interrupt the program during execution and calling service specified.

DIV − Used to divide the unsigned word by byte or unsigned double word by word.

ADD − Used to add the provided byte to byte/word to word.

LOOP – Used to loop a group of instructions until the condition satisfies.

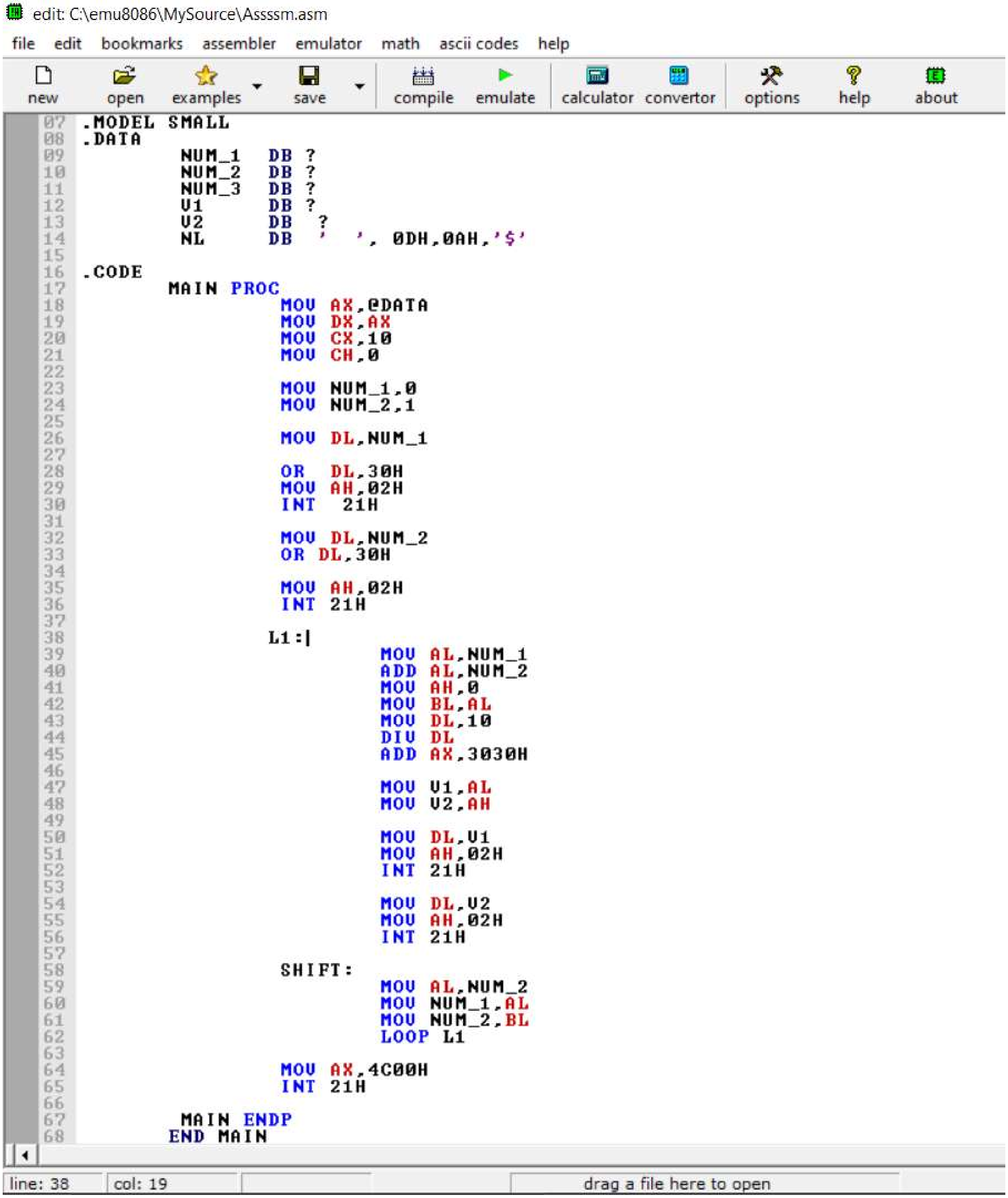
Program in Assembly:

```
.MODEL SMALL
.DATA
       NUM_1  DB ?
       NUM_2  DB ?
       NUM_3  DB ?
       V1     DB ?
       V2     DB  ?
       NL     DB  '  ', 0DH,0AH,'$'
.CODE
      MAIN PROC
               MOV AX,@DATA
               MOV DX,AX
               MOV CX,10
               MOV CH,0

               MOV NUM_1,0
               MOV NUM_2,1

               MOV DL,NUM_1

               OR  DL,30H
               MOV AH,02H
               INT  21H
```

```
   MOV DL,NUM_2
   OR DL,30H

   MOV AH,02H
   INT 21H
L1:

   MOV AL,NUM_1
   ADD AL,NUM_2
   MOV AH,0
   MOV BL,AL
   MOV DL,10
   DIV DL
   ADD AX,3030H

   MOV V1,AL
   MOV V2,AH

   MOV DL,V1
   MOV AH,02H
   INT 21H

   MOV DL,V2
   MOV AH,02H
   INT 21H
```

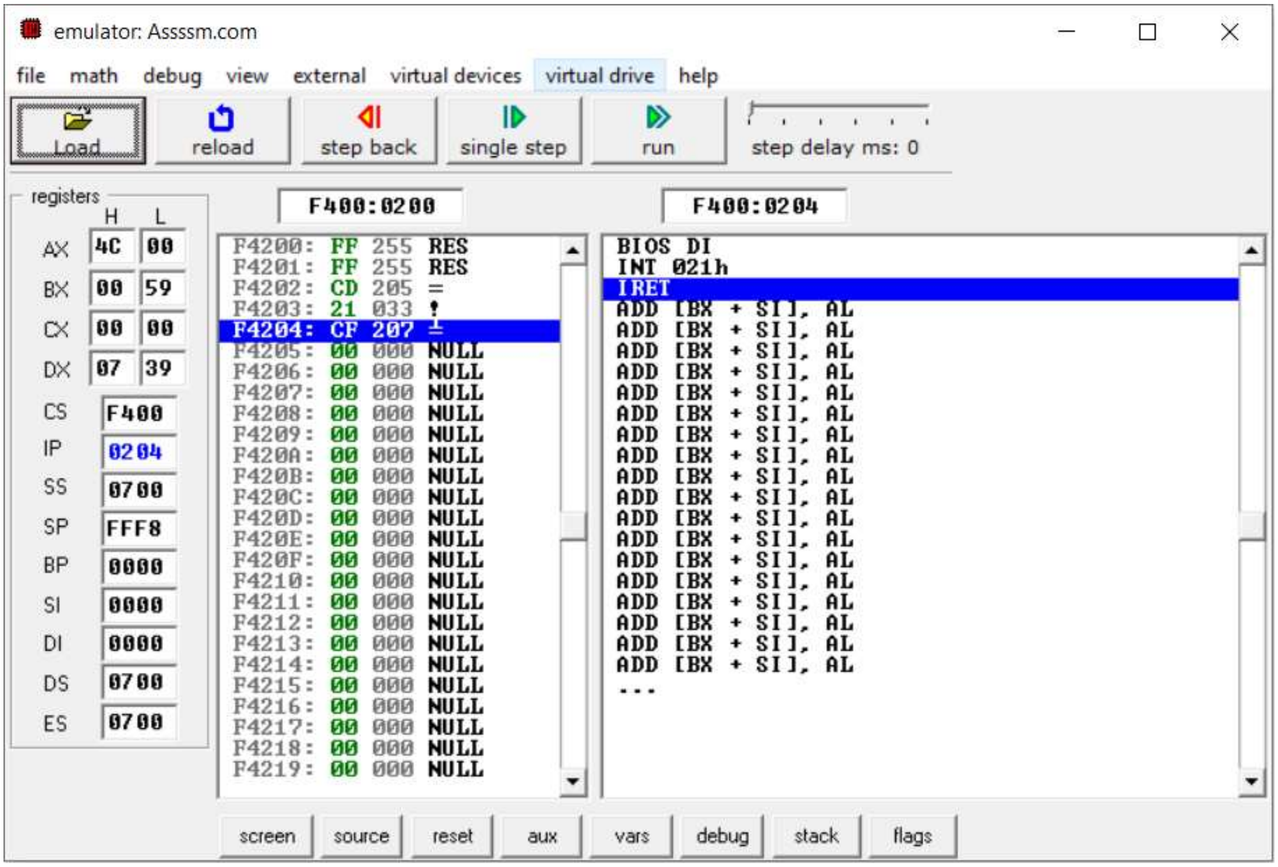
```
         SHIFT:
            MOV AL,NUM_2
            MOV NUM_1,AL
            MOV NUM_2,BL
            LOOP L1
            MOV AX,4C00H
            INT 21H

    MAIN ENDP
END MAIN
```

Code In 8086 Emulator:



```
.MODEL SMALL
.DATA
        NUM_1   DB ?
        NUM_2   DB ?
        NUM_3   DB ?
        V1      DB ?
        V2      DB  ?
        NL      DB  '  ', 0DH,0AH,'$'

.CODE
       MAIN PROC
                MOV AX,@DATA
                MOV DX,AX
                MOV CX,10
                MOV CH,0

                MOV NUM_1,0
                MOV NUM_2,1

                MOV DL,NUM_1

                OR  DL,30H
                MOV AH,02H
                INT  21H

                MOV DL,NUM_2
                OR DL,30H

                MOV AH,02H
                INT 21H

               L1:
                        MOV AL,NUM_1
                        ADD AL,NUM_2
                        MOV AH,0
                        MOV BL,AL
                        MOV DL,10
                        DIV DL
                        ADD AX,3030H

                        MOV V1,AL
                        MOV V2,AH

                        MOV DL,V1
                        MOV AH,02H
                        INT 21H

                        MOV DL,V2
                        MOV AH,02H
                        INT 21H

                SHIFT:
                        MOV AL,NUM_2
                        MOV NUM_1,AL
                        MOV NUM_2,BL
                        LOOP L1

                MOV AX,4C00H
                INT 21H

        MAIN ENDP
       END MAIN
```

Emulator:

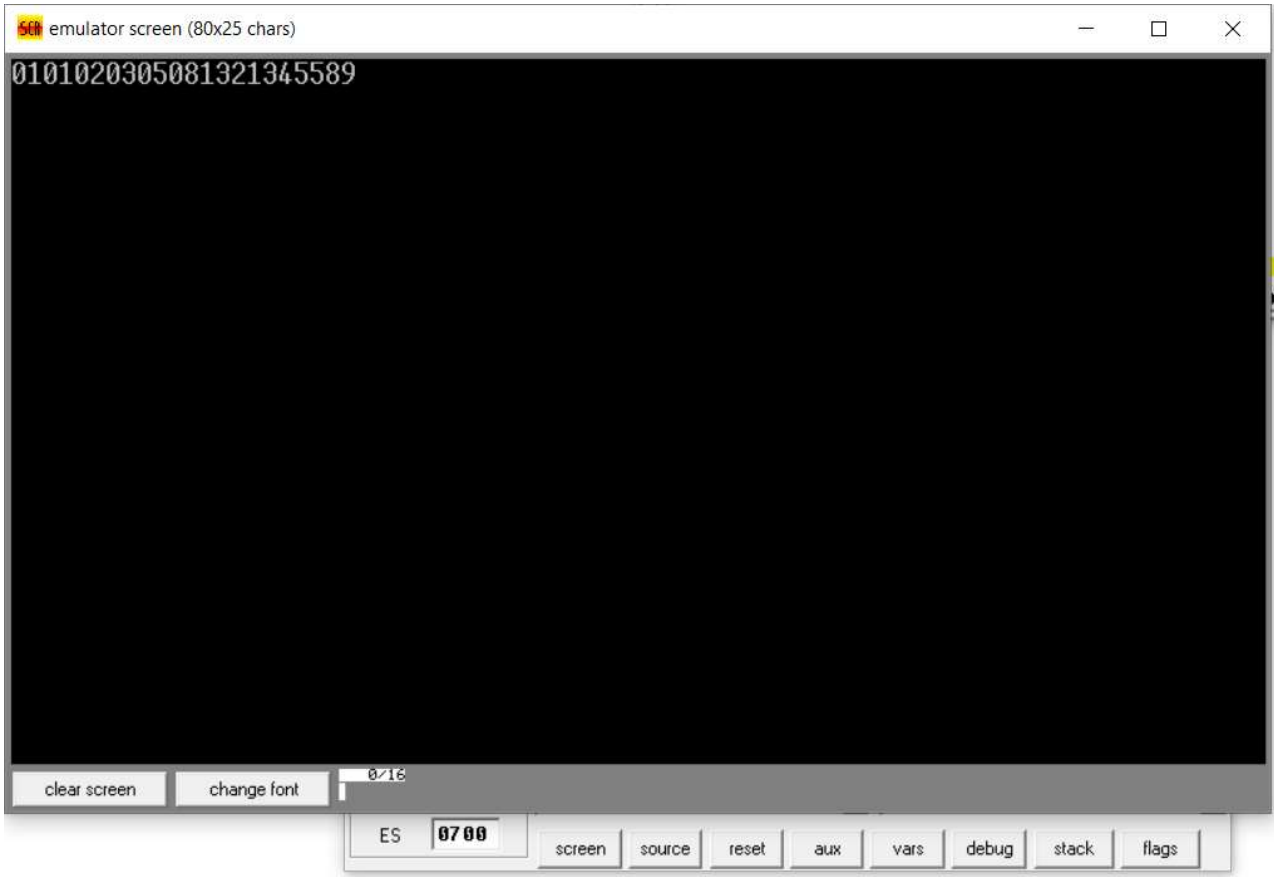


	H	L
AX	4C	00
BX	00	59
CX	00	00
DX	07	39
CS	F400	
IP	0204	
SS	0700	
SP	FFF8	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

```
F400:0200
F4200: FF 255 RES
F4201: FF 255 RES
F4202: CD 205 ═
F4203: 21 033 !
F4204: CF 207 ╧
F4205: 00 000 NULL
F4206: 00 000 NULL
F4207: 00 000 NULL
F4208: 00 000 NULL
F4209: 00 000 NULL
F420A: 00 000 NULL
F420B: 00 000 NULL
F420C: 00 000 NULL
F420D: 00 000 NULL
F420E: 00 000 NULL
F420F: 00 000 NULL
F4210: 00 000 NULL
F4211: 00 000 NULL
F4212: 00 000 NULL
F4213: 00 000 NULL
F4214: 00 000 NULL
F4215: 00 000 NULL
F4216: 00 000 NULL
F4217: 00 000 NULL
F4218: 00 000 NULL
F4219: 00 000 NULL
```

```
F400:0204
BIOS DI
INT 021h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

Output:



```
0101020305081321345589
```

Conclusion: Fibonacci Series up to 11 numbers is generated using Assembly Language.

Generated Output: 01 01 02 03 05 08 13 21 34 55 89

****