VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Experiment 4:

Fibonacci Series, Factorial Number, Permutation and Combination of a Number

Programme	:	**BTech. CSE Core**	Semester	:	**Win 2021-22**
Course	:	**Microprocessor and Interfacing**	Code	:	**CSE2006**
Faculty	:	**Dr. Florence Gnana Poovathy J**	Slot	:	**L15+L16**
Name	:	**Hariket Sukesh Kumar Sheth**	Register No.	:	**20BCE1975**

Date: 09-02-2022	Exp. 04	Fibonacci and Factorial Number	VIT Vellore Institute of Technology (Deemed to be University under section 3 of UGC Act, 1956)

Fibonnaci Numbers

Aim: To print the Fibonacci Series

Tool Used: Assembler – MASM611

Algorithm:

Step 1: First of all, mount the c drive using the command: **mount c c:\masm611\bin**

Step 2: After pressing **enter,** type **c:** and press enter.

Step 3: Now give a command, **fib.asm** for writing/editing the code and the write the code.

Step 4: A pop window appears; there we have to write out code(instructions) following the logic given below.

1. An array (DB) of 10 values is created, and stored in AX register.
2. SI register is pointed to the start of array. 10 is stored in CX register, and 0 is stored in AL register. The value stored in AL is moved to the start of array, pointed by [SI].
3. AL and SI registers are incremented by 1 each.
4. Again, the value stored in AL is moved to the start of array, pointed by [SI].
5. 0 is stored in AL, and then it heads into the loop L1.
6. Within the loop, the values stored in AL and the starting value of SI are added, and stored in AL register.
7. SI is incremented by 1, and the result of addition is moved from AL to SI register.
8. SI is then decremented by 1, and the value stored in [SI] is moved to AL register, and SI is again incremented.
9. MOVE AH, 4CH is used to interrupt the program, and HLT brings it a halt.

Step 5: Now give a command, **masm fib.asm** for running the code. The object file is created

Step 6: Now give a command, **link fib.obj** to link the object file to library file present in the bin folder.

Step 7: Press **ENTER** four times.

Step 8: Write debug **fib.exe**

-u

-g (followed by the **address of HLT or INT** to view the values in registers).

Program:

FIB.ASM

```
DATA SEGMENT
        FIB DB 10 DUP(0)
DATA ENDS

CODE SEGMENT
        ASSUME CS: CODE, DS: DATA

START:
        MOV AX, DATA
        MOV DS, AX
        MOV CX, 10
        LEA SI, FIB
        MOV AX, 0H
        MOV BX, 1H
        MOV [SI], AX
        INC SI
        MOV [SI], BX
        INC SI

L1:  ADD AX, BX
        MOV DX, AX
        MOV [SI], DX
        MOV AX, BX
        MOV BX, DX
        INC SI
        DEC CX
        JNZ L1
        HLT

CODE ENDS
END START
```

Sample Input:	**Sample Output:**
AX: 0H, BX: 1H CX: 10	Fibonacci Series: 0H, 1H, 1H, 2H, 3H, 5H, 8H, 0DH (13), 15H (21), 22H(34)

Register / Memory Contents for I/O:

```
C:\>debug fib.exe
-u
0765:0000 B86407        MOV     AX,0764
0765:0003 8ED8          MOV     DS,AX
0765:0005 B90A00        MOV     CX,000A
0765:0008 8D360000      LEA     SI,[0000]
0765:000C B80000        MOV     AX,0000
0765:000F BB0100        MOV     BX,0001
0765:0012 8904          MOV     [SI],AX
0765:0014 46            INC     SI
0765:0015 891C          MOV     [SI],BX
0765:0017 46            INC     SI
0765:0018 03C3          ADD     AX,BX
0765:001A 8BD0          MOV     DX,AX
0765:001C 8914          MOV     [SI],DX
0765:001E 8BC3          MOV     AX,BX
```

```
-u
0765:0020 8BDA          MOV     BX,DX
0765:0022 46            INC     SI
0765:0023 49            DEC     CX
0765:0024 75F2          JNZ     0018
0765:0026 F4            HLT
0765:0027 041C          ADD     AL,1C
0765:0029 0408          ADD     AL,08
0765:002B 44            INC     SP
0765:002C 68            DB      68
0765:002D 20C0          AND     AL,AL
0765:002F CAB320        RETF    20B3
0765:0032 741C          JZ      0050
0765:0034 04CE          ADD     AL,CE
0765:0036 9A1C041C20    CALL    201C:041C
0765:003B E504          IN      AX,04
0765:003D 0C44          OR      AL,44
0765:003F 64            DB      64
```

Output:

```
-g 0765:0026

AX=0037  BX=0059  CX=0000  DX=0059  SP=0000  BP=0000  SI=000C  DI=0000
DS=0764  ES=0754  SS=0763  CS=0765  IP=0026   NV UP EI PL ZR NA PE NC
0765:0026 F4            HLT
-d 0764: 0000 0009
0764:0000  00 01 01 02 03 05 08 0D-15 22                    .........."
```

Date: 09-02-2022	Exp. 04	Fibonacci and Factorial Number	VIT Vellore Institute of Technology (Deemed to be University under section 3 of UGC Act, 1956)

Factorial Number

Aim: To Perform Factorial of the Number

Tool Used: Assembler – MASM611

Algorithm:

Step 1: First of all, mount the c drive using the command: **mount c c:\masm611\bin**

Step 2: After pressing **enter,** type **c:** and press enter.

Step 3: Now give a command, **fact1975.asm** for writing/editing the code and the write the code.

Step 4: A pop window appears; there we have to write out code(instructions) following the logic given below.

1. 1H is stored in AX register, and 4H is stored in CX register.
2. Within the loop, the value in CX and AX registers are multiplied and stored in AX, and CX is decremented.
3. The process continues until CX is zero, and then the loop is exited.
4. The final product can be found in AX register.

Step 5: Now give a command, **masm fact1975.asm** for running the code. The object file is created.

Step 6: Now give a command, **link fact1975.obj** to link the object file to library file present in the bin folder.

Step 7: Press **ENTER** four times.

Step 8: Write debug **fact1975.exe**

-u

-g (followed by the **address of HLT or INT** to view the values in registers).

Program:

FACT1975.ASM

```
CODE SEGMENT
ASSUME CS: CODE

START:
        MOV AX, 1H
        MOV CX, 6H
LOOP1:
        MUL CX
        LOOP LOOP1
HLT
CODE ENDS
END START
```

Sample Input:	**Sample Output:**
Input: 6H	Result AX: 02D0H = 720

Register / Memory Contents for I/O:

```
C:\>debug fact1975.exe
-u
0764:0000 B80100        MOV     AX,0001
0764:0003 B90600        MOV     CX,0006
0764:0006 F7E1          MUL     CX
0764:0008 E2FC          LOOP    0006
0764:000A F4            HLT
0764:000B 1C68          SBB     AL,68
0764:000D 014070        ADD     [BX+SI+70],AX
0764:0010 1CEB          SBB     AL,EB
0764:0012 2C04          SUB     AL,04
0764:0014 1C04          SBB     AL,04
0764:0016 1C5D          SBB     AL,5D
0764:0018 9E            SAHF
0764:0019 7001          JO      001C
0764:001B 207B1C        AND     [BP+DI+1C],BH
0764:001E 75D6          JNZ     FFF6
-
```

Output:

```
-g 0764:000A

AX=02D0  BX=0000  CX=0000  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0754  ES=0754  SS=0763  CS=0764  IP=000A   NV UP EI PL NZ NA PO NC
0764:000A F4            HLT
```

Date: 09-02-2022	Exp. 05	Permutation and Combination	VIT Vellore Institute of Technology (Deemed to be University under section 3 of UGC Act, 1956)

Permutation

Aim: To find the Permutation of the number

Tool Used: Assembler – MASM611

Algorithm:

Step 1: First of all, mount the c drive using the command: **mount c c:\masm611\bin**

Step 2: After pressing **enter,** type **c:** and press enter.

Step 3: Now give a command, **permt.asm** for writing/editing the code and the write the code.

Step 4: A pop window appears; there we have to write out code(instructions) following the logic given below.

1. Initialize CX with N, AX with 1 and BX with r
2. Subtract r from N and store the result in CX
3. Store (N-r)! in AX
4. Load CX with N and move (N-r)! to BX
5. Store N! in AX
6. Divide AX by BX to store the final result in AX

Step 5: Now give a command, **masm permt.asm** for running the code. The object file is created.

Step 6: Now give a command, **link permt.obj** to link the object file to library file present in the bin folder.

Step 7: Press **ENTER** four times.

Step 8: Write debug **permt.exe**

-u

-g (followed by the **address of HLT or INT** to view the values in registers).

Program:

PERMT.ASM

```
CODE SEGMENT
ASSUME CS: CODE
START:
        MOV CX, 6H
        MOV AX, 1H
        MOV BX, 2H
        SUB CX, BX

LOOP1:
        MUL CX
        LOOP LOOP1
        MOV CX, 6H
        MOV BX, AX
        MOV AX, 1H
LOOP2:
        MUL CX
        LOOP LOOP2
        DIV BX
HLT
CODE ENDS
END START
```

Sample Input:	**Sample Output:**
Input: 6H, 2H ${}^{6}P_{2}$ = 6!/(6-2)!	Result AX: 001EH = 30

Register / Memory Contents for I/O:

```
C:\>debug permt.exe
-u
0764:0000 B90600        MOV     CX,0006
0764:0003 B80100        MOV     AX,0001
0764:0006 BB0200        MOV     BX,0002
0764:0009 2BCB          SUB     CX,BX
0764:000B F7E1          MUL     CX
0764:000D E2FC          LOOP    000B
0764:000F B90600        MOV     CX,0006
0764:0012 8BD8          MOV     BX,AX
0764:0014 B80100        MOV     AX,0001
0764:0017 F7E1          MUL     CX
0764:0019 E2FC          LOOP    0017
0764:001B F7F3          DIV     BX
0764:001D F4            HLT
0764:001E 75D6          JNZ     FFF6
```

Output:

```
-g 0764:001D

AX=001E  BX=0018  CX=0000  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0754  ES=0754  SS=0763  CS=0764  IP=001D   NV UP EI PL NZ NA PO NC
0764:001D F4            HLT
-
```

Date: 09-02-2022	Exp. 05	Permutation and Combination	VIT Vellore Institute of Technology (Deemed to be University under section 3 of UGC Act, 1956)

Combination

Aim: To find the Combination of the number

Tool Used: Assembler – MASM611

Algorithm:

Step 1: First of all, mount the c drive using the command: **mount c c:\masm611\bin**

Step 2: After pressing **enter,** type **c:** and press enter.

Step 3: Now give a command, **comb1975.asm** for writing/editing the code and the write the code.

Step 4: A pop window appears; there we have to write out code(instructions) following the logic given below.

1. Initialize CX with N, AX with 1 and BX with r
2. Subtract r from N and store the result in CX
3. Store (N-r)! in AX
4. Load CX with N and move (N-r)! to BX
5. Store N! in AX
6. Divide AX by BX
7. Move r into CX
8. Move the present value of AX which is N!/(N-r)! into BX
9. Store r! in AX
10. 10.Exchange the values of AX and BX.
11. Divide AX by BX to obtain the final result

Step 5: Now give a command, **masm comb1975.asm** for running the code. The object file is created.

Step 6: Now give a command, **link comb1975.obj** to link the object file to library file present in the bin folder.

Step 7: Press **ENTER** four times.

Step 8: Write debug **comb1975.exe**

-u

-g (followed by the **address of HLT or INT** to view the values in registers).

Program:

COMB1975.ASM

```
CODE SEGMENT
ASSUME CS: CODE
START:
        MOV CX, 6H
        MOV AX, 1H
        MOV BX, 2H
        SUB CX, BX

LOOP1:
        MUL CX
        LOOP LOOP1
        MOV CX, 6H
        MOV BX, AX
        MOV AX, 1H
LOOP2:
        MUL CX
        LOOP LOOP2
        DIV BX
        MOV CX, 2H
        MOV BX, AX
        MOV AX, 1H
LOOP3:
        MUL CX
        LOOP LOOP3
        XCHG AX,BX
        DIV BX
HLT
CODE ENDS
END START
```

Sample Input:	**Sample Output:**
Input: 6H, 2H ${}^{6}C_{2}$ = 6!/(6-2)!*2!	Result AX: 000FH = 15

Register / Memory Contents for I/O:

```
C:\>debug comb1975.exe
-u
0764:0000 B90600        MOV     CX,0006
0764:0003 B80100        MOV     AX,0001
0764:0006 BB0200        MOV     BX,0002
0764:0009 2BCB          SUB     CX,BX
0764:000B F7E1          MUL     CX
0764:000D E2FC          LOOP    000B
0764:000F B90600        MOV     CX,0006
0764:0012 8BD8          MOV     BX,AX
0764:0014 B80100        MOV     AX,0001
0764:0017 F7E1          MUL     CX
0764:0019 E2FC          LOOP    0017
0764:001B F7F3          DIV     BX
0764:001D B90200        MOV     CX,0002
-_
```

```
-u
0764:0020 8BD8          MOV     BX,AX
0764:0022 B80100        MOV     AX,0001
0764:0025 F7E1          MUL     CX
0764:0027 E2FC          LOOP    0025
0764:0029 93            XCHG    BX,AX
0764:002A F7F3          DIV     BX
0764:002C F4            HLT
0764:002D 7D22          JGE     0051
0764:002F 6C            DB      6C
0764:0030 67            DB      67
0764:0031 9D            POPF
0764:0032 1C04          SBB     AL,04
0764:0034 1C35          SBB     AL,35
0764:0036 41            INC     CX
0764:0037 041C          ADD     AL,1C
0764:0039 0408          ADD     AL,08
0764:003B 44            INC     SP
0764:003C 68            DB      68
0764:003D 20C0          AND     AL,AL
0764:003F CAB320        RETF    20B3
```

Output:

```
-g 0764:002C

AX=000F  BX=0002  CX=0000  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0754  ES=0754  SS=0763  CS=0764  IP=002C   NV UP EI PL NZ NA PO NC
0764:002C F4            HLT
-
```