



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

Experiment 9:

Basic Programming with 8087

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: 23-03-2022

Exp. 09

Basic Programming
with 8087**VIT**
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Basic Programming with 8087

Aim: To calculate the Volume of Cone

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, **<filename>.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.

- (i) Four variables are defined, one to hold the value of the radius, another to hold the value of the constant $1/3$, another one to store the value of height and the last one to store the value of the sphere's volume after calculation.
- (ii) The data is moved to DS register, via AX register.
- (iii) FINIT signals the initialization of 8087 commands and registers.
- (iv) The variable RAD is loaded onto stack top ST(0) via FLD.
- (v) A copy of it is stored in ST(4) via FST.
- (vi) The values stored in ST(0) and ST(4) are multiplied once, and stored at ST(0).
- (vii) The variable CONS is loaded onto stack top via FLD, while the value previously stored at ST(0) gets pushed to ST(1).
- (viii) Similarly HEIGHT variable value is loaded and multiplied
- (ix) The values in ST(1) and ST(0) are multiplied via FMUL, and stored at ST(0).
- (x) π is loaded onto stack top via FLDPI, and the already present values at ST(0) and ST(1) are pushed to ST(1) and ST(2) respectively.
- (xi) The values in ST(1) and ST(0) are multiplied by FMUL, and stored at ST(0).
- (xii) The calculated volume is then stored in the variable VOL via FST.

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

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

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

Step 8: Write **<filename>.exe**

PROGRAM:

```

                                CONE1.ASM
DATA SEGMENT
ORG 1000h
    RADIUS DD 2.9061
    HEIGHT DD 4.5
    CONST DD 0.3333
    VOLUME DD 01 DUP(?)
DATA ENDS

CODE SEGMENT
ASSUME CS: CODE, DS:DATA
.8087
START:
    MOV AX, DATA
    MOV DS, AX
    FINIT
    FLD RADIUS
    FST ST(4)
    FMUL ST(0), ST(4)
    FLD CONST
    FMUL
    FLDP1

    FMUL
    FLD HEIGHT
    FMUL
    FST VOLUME
    HLT
    CODE ENDS
END START
END

```

OUTPUT:

```

C:\>debug CONE1.exe
-u
0065:0000 B86407      MOV     AX,0764
0065:0003 8ED8        MOV     DS,AX
0065:0005 9B          WAIT
0065:0006 DBE3        FINIT
0065:0008 9B          WAIT
0065:0009 D9060010     FLD     DWORD PTR [1000]
0065:000D 9B          WAIT
0065:000E DDD4        FST     ST(4)
0065:0010 9B          WAIT
0065:0011 D8CC        FMUL    ST,ST(4)
0065:0013 9B          WAIT
0065:0014 D9060010     FLD     DWORD PTR [1000]
0065:0018 9B          WAIT
0065:0019 DEC9        FMULP   ST(1),ST
0065:001B 9B          WAIT
0065:001C D9EB        FLDP1
0065:001E 9B          WAIT
0065:001F DEC9        FMULP   ST(1),ST

```

```

-u
0065:0021 9B      WAIT
0065:0022 D9060410    FLD     DWORD PTR [1004]
0065:0026 9B      WAIT
0065:0027 DEC9      FMULP   ST(1),ST
0065:0029 9B      WAIT
0065:002A D9160C10    FST     DWORD PTR [100C]
0065:002E F4      HLT
0065:002F 0000      ADD     [BX+SI],AL
0065:0031 0000      ADD     [BX+SI],AL
0065:0033 0000      ADD     [BX+SI],AL
0065:0035 0000      ADD     [BX+SI],AL
0065:0037 0000      ADD     [BX+SI],AL
0065:0039 0000      ADD     [BX+SI],AL
0065:003B 0000      ADD     [BX+SI],AL
0065:003D 0000      ADD     [BX+SI],AL
0065:003F 0000      ADD     [BX+SI],AL

```

REGISTER / MEMORY CONTENTS

```

-g 0065:002E
AX=0764 BX=0000 CX=103F DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0764 ES=0754 SS=0763 CS=0065 IP=002E  NU UP EI PL NZ NA PO NC
0065:002E F4      HLT
-d 0764:1000 100B
0764:1000 8B FD 39 40 00 00 90 40-4C A6 AA 3E      ..90...0L...>
0764:1010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
-d 0764:100B 100F
0764:1000      3E 2C 2D 1F 42      >,-.B

```

We would get the result by reversing this
Actual = 42 1F 2D 2C

2C 2D 1F 42

Sample Input	Sample Output
Radius = 2.9061 Height = 4.5	Hexadecimal Result = 2C 2D 1F 42 Actual Result = 42 1F 2D 2C Result = 39.7941