**Practical - 24**

**Aim:** Write an assembly code to arrange elements of a vector in descending order. The program must have subroutine to perform sorting task.

**Description of instructions used:**

**Subroutines:**  a subroutine is a sequence of program instructions that perform a specific task, packaged as a unit. This unit can then be used in programs wherever that particular task have to be performed. A subroutine is often coded so that it can be started (called) several times and from several places during one execution of the program, including from other subroutines, and then branch back (return) to the next instruction after the call, once the subroutine’s task is done. It is implemented by using Call and Return instructions.

**DB** **(DEFINE BYTE):** The **DB** directive is used to declare a byte type variable, or a set aside one or more storage locations of type byte in memory.

**REVERSE PROC NEAR:** This identifies the start of a procedure named REVERSE and tells the assembler that the procedure is far (in a segment with different name from the one that contains the instructions which calls the procedure). The PROC directive is used with the ENDP directive to “bracket” a procedure.

* **NEAR:** the procedure resides in the same code segment. (Local)
* **FAR:** resides at any location in the memory.

**RET:** The **RET** instruction stands for return. This instruction is used at the end of the procedures or the subprograms. This instruction transfers the execution to the caller program.

**CALL:** The **CALL instruction** is used whenever we need to make a call to some procedure or a subprogram.

**DEC:** Used to decrement the provided byte/word by 1.

**INC:** Used to increment the provided byte/word by 1.

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

**JMP:** the JMP instruction performs an unconditional jump. Such an instruction transfers the flow of execution by changing the instruction pointer register.

**Code:**

DATA SEGMENT

STRING1 DB 90H,10H,36H,95H,66H

DATA ENDS

CODE SEGMENT

REVERSE PROC NEAR

ASSUME CS:CODE,DS:DATA

UP2:

MOV CL,04H

LEA SI,STRING1

UP1:

MOV AL,[SI]

MOV BL,[SI+1]

CMP AL,BL

JNC DOWN

MOV DL,[SI+1]

XCHG [SI],DL

MOV [SI+1],DL

DOWN:

INC SI

DEC CL

JNZ UP1

DEC CH

JNZ UP2

INT 3

RET

REVERSE ENDP

START:

MOV AX,DATA

MOV DS,AX

MOV CH,04H

CALL REVERSE

CODE ENDS

END START

**Output:**

