

1.1 Addition

```
CODE SEGMENT
ASSUME CS: CODE
START:
MOV AX,2345H
MOV BX,1234H
ADD AX,BX
HLT
CODE ENDS
END START
```

1.3 Multiplication

```
CODE SEGMENT
ASSUME CS: CODE
START:
MOV AX,12H
MOV BX,12H
MUL BX
HLT
CODE ENDS
END START
```

2. Sum of series

```
DATA SEGMENT
A DB 1,2,3,4,5,6,7,8,9,10
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:
MOV AX,DATA
MOV DS,AX
MOV CL,10
LEA SI,A
MOV AH,00
MOV AL,00
L1:
ADD AL,[SI]
INC SI
DEC CL
CMP CL,00
JNZ L1
MOV AH,4CH
INT 21H
CODE ENDS
END START
```

1.2 Subtraction

```
CODE SEGMENT
ASSUME CS: CODE
START:
MOV AX,2345H
MOV BX,1234H
SUB AX,BX
HLT
CODE ENDS
END START
```

1.4 Division

```
CODE SEGMENT
ASSUME CS: CODE
START:
MOV AX,100
MOV BX,2
DIV BX
HLT
CODE ENDS
END START
```

3.1 Smallest

```
DATA SEGMENT
STR1 DB 99H,01H,32H,47H,73H
RESULT DB 0
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:
MOV AX,DATA
MOV DS,AX
MOV CL,05H
DEC CL
LEA SI,STR1
MOV AL,[SI]
LOC1:
CMP AL,[SI+1]
JB LOC2
MOV AL,[SI+1]
LOC2:
INC SI
DEC CL
JNZ LOC1
MOV RESULT,AL
MOV AH,4CH
INT 21H
CODE ENDS
END START
```

3.2 Largest

```
DATA SEGMENT
STR1 DB 99H,01H,32H,50H,47H
RESULT DB 0
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:
MOV AX,DATA
MOV DS,AX
MOV CL,05H
DEC CL
LEA SI,STR1
MOV AL,[SI]
LOC1:
CMP AL,[SI+1]
JA LOC2;IF AL>[SI+1] IT WILL JUMP TO LOC2
MOV AL,[SI+1]
LOC2:
INC SI
DEC CL
JNZ LOC1;JUMPS TO LOC1 IF CL!=0
MOV RESULT,AL
MOV AH,4CH
INT 21H
CODE ENDS
END START
```

4.1 Ascending

```
DATA SEGMENT
STR1 DB 99H, 1H,12H, 47H,35H
DATA ENDS
CODE SEGMENT
ASSUME DS:DATA,CS:CODE
START:
MOV AX, DATA
MOV DS, AX
MOV CH, 04H
UP2:
MOV CL, 04H
LEA SI, STR1
UP1:
MOV AL, [SI]
MOV BL, [SI+1]
CMP AL, BL
JC DOWN
MOV DL, [SI+1]
XCHG [SI], DL
MOV [SI+1], DL
DOWN:
INC SI
DEC CL
JNZ UP1
DEC CH
JNZ UP2
MOV AH, 4CH
INT 21H
CODE ENDS
END START
```

4.2 Descending

```
DATA SEGMENT
STR1 DB 99H,12H,40H,72H,36H
DATA ENDS
CODE SEGMENT
ASSUME DS:DATA,CS:CODE
START:
MOV AX,DATA
MOV DS,AX
MOV CH,04H
UP2:
MOV CL,04H
LEA SI,STR1
UP1:
MOV AL,[SI]
MOV BL,[SI+1]
CMP AL,BL
JAE DOWN
MOV DL,[SI+1]
XCHG [SI],DL
MOV [SI+1],DL
DOWN:
INC SI
DEC CL
JNZ UP1
DEC CH
JNZ UP2
MOV AH,4CH
INT 21H
CODE ENDS
END START
```

5. 1 ODD AND EVEN

```
DATA SEGMENT
N DW 11H
MSG1 DB "ODD$"
MSG2 DB "EVEN$"
DATA ENDS
PRINT MACRO MSG
MOV AH,09H
LEA DX,MSG
INT 21H
ENDM
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:
MOV AX,DATA
MOV DS,AX
MOV AX,N
TEST AL,01H
JZ EVEN
PRINT MSG1
JMP LAST
EVEN:
PRINT MSG2
LAST:
MOV AH,4CH
INT 21H
CODE ENDS
END START
```

5.2 Positive or negative

```
DATA SEGMENT  
N DW 12H  
MSG1 DB "POSITIVE$"  
MSG2 DB "NEGATIVE$"  
DATA ENDS
```

```
PRINT MACRO MSG  
MOV AH,09H  
LEA DX,MSG  
INT 21H  
MOV AH,4CH  
INT 21H  
ENDM
```

```
CODE SEGMENT  
ASSUME CS:CODE,DS:DATA
```

```
START:  
MOV AX,DATA  
MOV DS,AX  
MOV AX,N  
ROL AX,1  
JNC POS  
PRINT MSG2  
JMP LAST  
POS:  
PRINT MSG1  
LAST:  
MOV AH,4CH  
INT 21H  
  
CODE ENDS  
END START
```

6. Block transfer

```
DATA SEGMENT  
STR1 DB 01H,02H,03H,04H,05H  
STR2 DB 5 DUP(0)  
DATA ENDS
```

```
CODE SEGMENT  
ASSUME CS:CODE,DS:DATA  
START:  
MOV AX,DATA  
MOV DS,AX  
MOV ES,AX  
LEA SI,STR1  
LEA DI,STR2  
MOV CX,05H  
CLD  
REP MOVSB  
MOV AH,4CH  
INT 21H  
CODE ENDS
```

7. String length

```
DATA SEGMENT  
SIG DB "MICROPROCESSOR$"  
LEN DB 0  
DATA ENDS
```

```
CODE SEGMENT  
ASSUME CS:CODE,DS:DATA,ES:EXTRA  
START:  
MOV AX,DATA  
MOV DS,AX  
LEA DI,SIG  
MOV AL,24H  
MOV BL,00H  
DO:  
INC BL  
SCASB  
JNZ DO  
JMP DONE  
DONE:  
DEC BL  
MOV LEN,BL  
MOV AH,4CH  
INT 21H  
CODE ENDS  
END START
```

8. String reverse

```
DATA SEGMENT
STR1 DB 01H, 02H, 03H, 04H, 05H
STR2 DB 5 DUP(0)
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START:
MOV AX, DATA
MOV DS, AX
LEA SI, STR1
LEA DI, STR2+4
MOV CX, 05H
BACK:
CLD
MOV AL, [SI]
MOV [DI], AL
INC SI
DEC DI
JNZ BACK
MOV AH,4CH
INT 21H
CODE ENDS
END START
```

9. Palindrome

```
DATA SEGMENT
BLOCK1 DB 'MALAYALAM'
MSG1 DB "IT IS PALINDROME$"
MSG2 DB "IT IS NOT A PALINDROME$"
PAL DB 00H
DATA ENDS
PRINT MACRO MSG
MOV AH,09H
LEA DX,MSG
INT 21H
MOV AX,4CH
INT 21H
ENDM
EXTRA SEGMENT
BLOCK2 DB 9 DUP(?)
EXTRA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA,ES:EXTRA
START:MOV AX,DATA
MOV DS,AX
MOV AX,EXTRA
MOV ES,AX
LEA SI,BLOCK1
LEA DI,BLOCK2+8
MOV CX,00009H
BACK:
CLD
LODSB
STD
STOSB
LOOP BACK
LEA SI,BLOCK1
LEA DI,BLOCK2
MOV CX,0009H
CLD
REPZ CMPSB
JNZ SKIP
PRINT MSG1
SKIP:PRINT MSG2
CODE ENDS
END START
```

10. Concatenation

```
DATA SEGMENT
STR1 DB "HELLO$"
STR2 DB "WORLD$"
DATA ENDS
CODE SEGMENT
ASSUME DS:DATA,CS:CODE
START:
MOV AX,DATA
MOV DS,AX
MOV SI,OFFSET STR1
NEXT:
MOV AL,[SI]
CMP AL,"$"
JE EXIT
INC SI
JMP NEXT
EXIT:
MOV DI,OFFSET STR2
UP:
MOV AL,[DI]
CMP AL,"$"
JE EXIT1
MOV [SI],AL
INC SI
INC DI
JMP UP
EXIT1:
MOV AL,"$"
MOV [SI],AL
MOV AH,4CH
INT 21H
CODE ENDS
END START
```

11. To count number of '0' and '1'

```
DATA SEGMENT
NO DW 3H
Z DW ?
O DW ?
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:
MOV AX,DATA
MOV DS,AX
MOV AX,NO
MOV BX,00H
MOV CX,10H
MOV DX,00H
UP:
ROL AX,1
JC ONE
INC BX
JMP NEXT
ONE:
INC DX
NEXT:
DEC CX
JNZ UP
MOV Z,BX
MOV O,DX
MOV AH,4CH
INT 21H
CODE ENDS
END START
```

