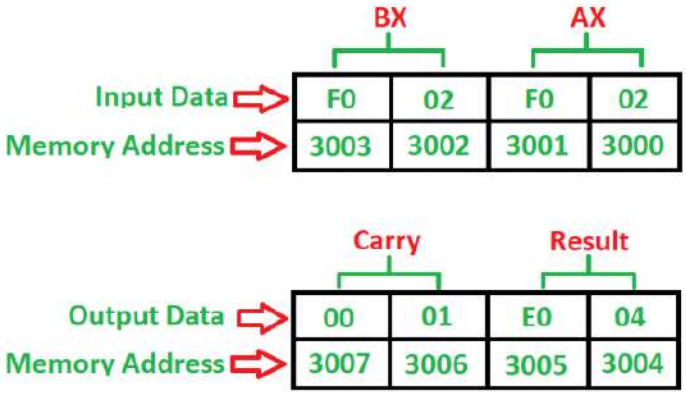
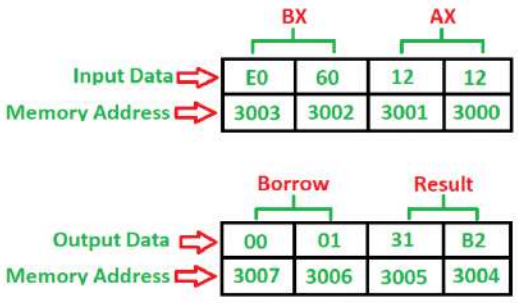
8086 Programs

Add two 16 bit numbers given by the user.



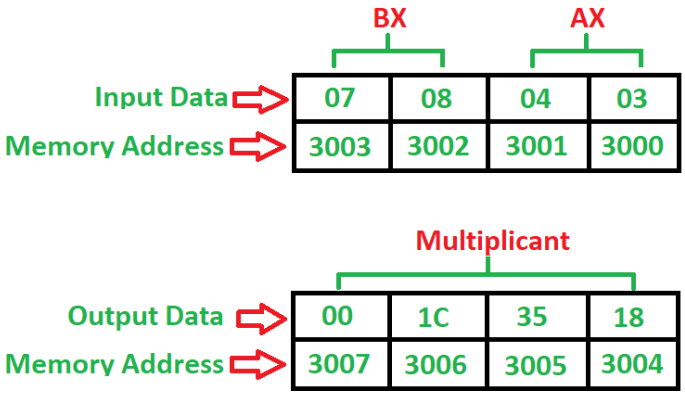
Memory	Mnemonics	Operands	Comment
2000	MOV	CX, 0000	[CX] <- 0000
2003	MOV	AX, [3000]	[AX] <- [3000]
2007	MOV	BX, [3002]	[BX] <- [3002]
200B	ADD	AX, BX	[AX] <- [AX] + [BX]
200D	JNC	2010	Jump if no carry
200F	INC	CX	[CX] <- [CX] + 1
2010	MOV	[3004], AX	[3004] <- [AX]
2014	MOV	[3006], CX	[3006] <- [CX]
2018	HLT		Stop

Subtract two 16 bit numbers given by the user.



Memory	Mnemonics	Operands	Comment
2000	MOV	CX, 0000	[CX] <- 0000
2003	MOV	AX, [3000]	[AX] <- [3000]
2007	MOV	BX, [3002]	[BX] <- [3002]
200B	SUB	AX, BX	[AX] <- [AX] – [BX]
200D	JNC	2010	Jump if no borrow
200F	INC	CX	[CX] <- [CX] + 1
2010	MOV	[3004], AX	[3004] <- [AX]
2014	MOV	[3006], CX	[3006] <- [CX]
2018	HLT		Stop

Multiply two 16 bit numbers given by the user.

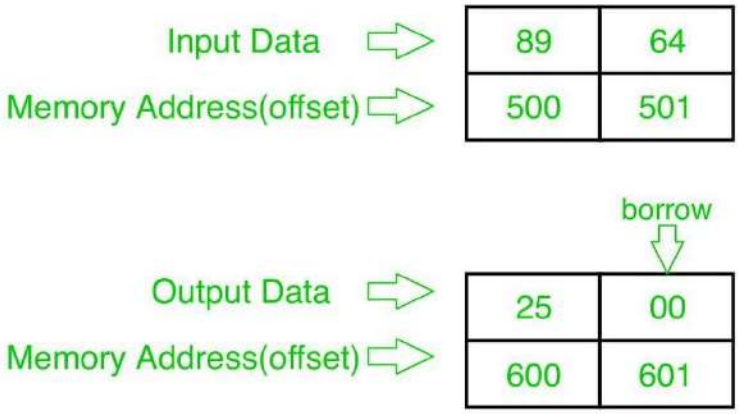


Memory	Mnemonics	Operands	Comment
2000	MOV	AX, [3000]	[AX] <- [3000]
2004	MOV	BX, [3002]	[BX] <- [3002]
2008	MUL	BX	[AX] <- [AX] * [BX]
200A	MOV	[3004], AX	[3004] <- AX
200E	MOV	AX, DX	[AX] <- [DX]
2010	MOV	[3006], AX	[3006] <- AX
2014	HLT		Stop

Divide 32 bit data by 16 bit data.

Address	Label	Mnemonics	Comments
01000		MOV AX, 4000H	[AX] =4000H
01003		MOV DS, AX	[DS] =4000H
01005		MOV AX, [1200H]	Get lower 16-bits of dividend input data in AX register. [AL] =[41200H] [AH] =[41201H]
01008		MOV DX, [1202H]	Get lower 16-bits of dividend input data in DX register. [DL] =[41202H] [DH] =[41203H]
0100C		MOV BX, [1204H]	Get 16-bit divisor input data in BX register. [BL] =[41204H] [BH] =[41205H]
01010		DIV BX	Divide the contents of registers (DX and AX) by contents of register BX.
01012		MOV [2000H], AX	Store the 16-bits of quotient from AX to specific memory locations. [AL] → [42000] [AH] → [42001]
01015		MOV AX, DX	Copy the 16-bits of remainder from DX register to AX register. [DX] → [AX]
01017		MOV [2002H], AX	Store the 16-bits of remainder from AX to specific memory locations. [AL] → [42002] [AH] → [42003]
0101A		HLT	Stop the program

Subtract two 8bit BCD numbers



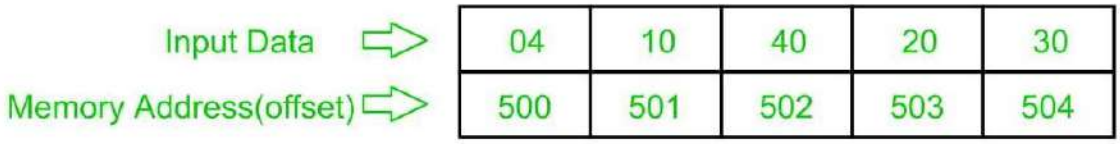
MEMORY ADDRESS	MNEMONICS	COMMENT
400	MOV AL, [500]	AL<-[500]
404	MOV BL, [501]	BL<-[501]
408	SUB AL, BL	AL<-AL-BL
40A	DAS	DECIMAL ADJUST AL

MEMORY ADDRESS	MNEMONICS	COMMENT
40B	MOV [600], AL	AL->[600]
40F	MOV AL, 00	AL<-00
411	ADC AL, AL	AL<-AL+AL+cy(prev)
413	MOV [601], AL	AL->[601]
417	HLT	END

Add two 8bit BCD numbers

MEMORY ADDRESS	MNEMONICS	COMMENT
400	MOV AL, [500]	AL<-[500]
404	MOV BL, [501]	BL<-[501]
408	ADD AL, BL	AL<-AL+BL
40A	DAA	DECIMAL ADJUST AL
40B	MOV [600], AL	AL->[600]
40F	MOV AL, 00	AL<-00
411	ADC AL, AL	AL<-AL+AL+cy(prev)
413	MOV [601], AL	AL->[601]
417	HLT	END

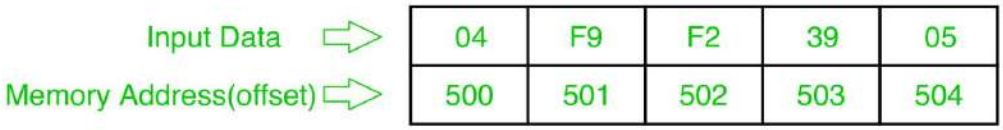
find the largest number from an array of n numbers stored in an array

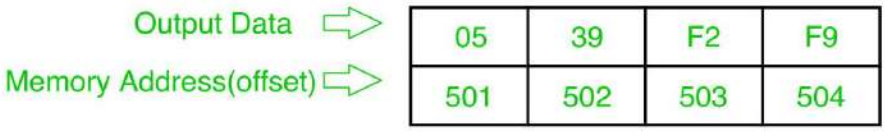


MEMORY ADDRESS	MNEMONICS	COMMENT
400	MOV SI, 500	SI<-500
403	MOV CL, [SI]	CL<-[SI]
405	MOV CH, 00	CH<-00
407	INC SI	SI<-SI+1
408	MOV AL, [SI]	AL<-[SI]
40A	DEC CL	CL<-CL-1
40C	INC SI	SI<-SI+1
40D	CMP AL, [SI]	AL-[SI]
40F	JNC 413	JUMP TO 413 IF CY=0
411	MOV AL, [SI]	AL<-[SI]
413	INC SI	SI<-SI+1
414	LOOP 40D	CX<-CX-1 & JUMP TO 40D IF CX NOT 0
416	MOV [600], AL	AL->[600]

MEMORY ADDRESS	MNEMONICS	COMMENT
41A	HLT	END

sort an integer array in ascending order





MEMORY ADDRESS	MNEMONICS	COMMENT
400	MOV SI, 500	SI<-500
403	MOV CL, [SI]	CL<-[SI]
405	DEC CL	CL<-CL-1
407	MOV SI, 500	SI<-500
40A	MOV CH, [SI]	CH<-[SI]
40C	DEC CH	CH<-CH-1
40E	INC SI	SI<-SI+1
40F	MOV AL, [SI]	AL<-[SI]
411	INC SI	SI<-SI+1
412	CMP AL, [SI]	AL-[SI]
414	JC 41C	JUMP TO 41C IF CY=1
416	XCHG AL, [SI]	SWAP AL AND [SI]

MEMORY ADDRESS	MNEMONICS	COMMENT
418	DEC SI	SI<-SI-1
419	XCHG AL, [SI]	SWAP AL AND [SI]
41B	INC SI	SI<-SI+1
41C	DEC CH	CH<-CH-1
41E	JNZ 40F	JUMP TO 40F IF ZF=0
420	DEC CL	CL<-CL-1
422	JNZ 407	JUMP TO 407 IF ZF=0
424	HLT	END

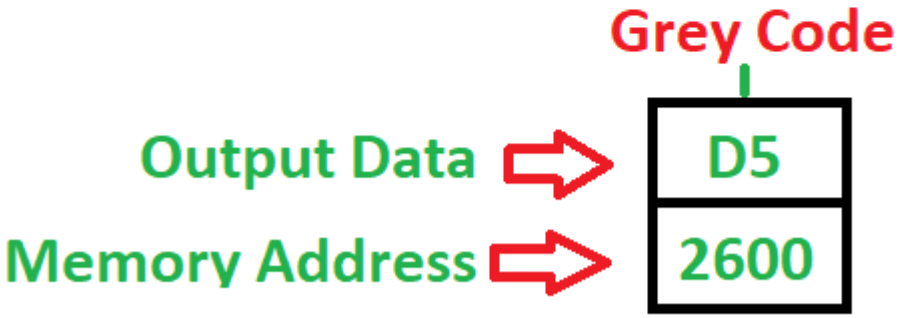
convert BCD number to its ASCII code equivalent.

```
Input:
DATA: 98H in memory location 2000

Output:
DATA: 38H in memory location 3000 and
      39H in memory location 3001
```

Memory Address	Mnemonics	Comments
400	MOV AL, [2000]	AL<-[2000]
404	MOV AH, AL	AH<-AL
406	AND AL, 0F	AL <- (AL AND 0F)
408	MOV CL, 04	CL <- 04
40A	SHR AH, CL	Shift AH content Right by 4 bits(value of CL)
40C	OR AX, 3030	AX <- (AX OR 3030)
40F	MOV [3000], AX	[3000]<-AX
413	HLT	Stop Execution

Binary number to its Gray code equivalent.



Memory	Mnemonics	Operands	Comment
2000	MOV	AL, [2500]	[AL] <- [2500]
2004	MOV	BL, AL	[BL] <- [AL]
2006	SHR	AL, 01	Shift Right one time
2008	XOR	BL, AL	[BL] <- [BL] @ AL
200A	MOV	[2600], BL	[2600] <- [BL]
200E	HLT		Stop