Task 01:

Take a number in AX, and if it's a negative number, replace it by 5.

```
MOV AX, -2       ; Load test value
CMP AX, 0
JGE DONE         ; If AX >= 0, skip replacement
MOV AX, 5        ; Replace AX with 5 if negative
DONE:
```

Task 02:

AL and BL contain ASCII. Display the one that comes first alphabetically.

```
MOV AL, 'D'
MOV BL, 'B'
CMP AL, BL
JL PRINT_AL      ; If AL < BL, print AL
MOV DL, BL
JMP DISPLAY

PRINT_AL:
MOV DL, AL

DISPLAY:
MOV AH, 2
INT 21H
```

Task 03:

If AX < 0, BX = -1; if AX = 0, BX = 0; if AX > 0, BX = 1

```
MOV AX, -5
CMP AX, 0
JL LESS
JE EQUAL
; Greater
MOV BX, 1
JMP END

LESS:
MOV BX, -1
```

```
JMP END

EQUAL:
MOV BX, 0

END:
```

Task 04:

If AL = 1 or 3 → print "o", if 2 or 4 → print "e"

```
MOV AL, 3
CMP AL, 1
JE PRINT_O
CMP AL, 3
JE PRINT_O
CMP AL, 2
JE PRINT_E
CMP AL, 4
JE PRINT_E
JMP END

PRINT_O:
MOV DL, 'o'
JMP PRINT

PRINT_E:
MOV DL, 'e'

PRINT:
MOV AH, 2
INT 21H

END:
```

Task 05:

Read a character, if uppercase (A–Z), print it

```
MOV AH, 1
INT 21H        ; Read char into AL

CMP AL, 'A'
JL END         ; Not uppercase
CMP AL, 'Z'
JG END

MOV DL, AL     ; Valid uppercase
MOV AH, 2
INT 21H

END:
```

Task 06:

Read char. If 'y' or 'Y', display. Else exit.

```
MOV AH, 1
INT 21H
CMP AL, 'y'
JE SHOW
CMP AL, 'Y'
JE SHOW
JMP END

SHOW:
MOV DL, AL
MOV AH, 2
INT 21H

END:
```

Task 07:

Check even or odd

```
MOV AL, 7
MOV AH, 0
MOV BL, 2
DIV BL          ; AL mod BL is in AH

CMP AH, 0
JE EVEN
MOV DL, 'O'     ; Odd
JMP PRINT

EVEN:
MOV DL, 'E'

PRINT:
MOV AH, 2
INT 21H
```

Task 08:

Check if input is vowel or consonant

```
MOV AH, 1
INT 21H

MOV BL, AL
CMP BL, 'A'
JE VOWEL
CMP BL, 'E'
JE VOWEL
CMP BL, 'I'
JE VOWEL
CMP BL, 'O'
JE VOWEL
CMP BL, 'U'
JE VOWEL
CMP BL, 'a'
JE VOWEL
```

```
CMP BL, 'e'
JE VOWEL
CMP BL, 'i'
JE VOWEL
CMP BL, 'o'
JE VOWEL
CMP BL, 'u'
JE VOWEL
JMP CONSONANT

VOWEL:
MOV DL, 'V'
JMP PRINT

CONSONANT:
MOV DL, 'C'

PRINT:
MOV AH, 2
INT 21H
```

Task 09:

Write a program to check whether a number is divisible by 5 and 11.

```
MOV AX, 55      ; Test value
MOV BX, 5
MOV DX, 0       ; Clear remainder
DIV BX          ; AX / 5 → result in AX, remainder in DX
CMP DX, 0
JNE NOT_DIV     ; If not divisible by 5, skip

MOV AX, 55      ; Reload original value
MOV BX, 11
MOV DX, 0
DIV BX
CMP DX, 0
JNE NOT_DIV

MOV DL, 'Y'     ; Divisible by both
JMP PRINT

NOT_DIV:
```

```
MOV DL, 'N'

PRINT:
MOV AH, 2
INT 21H
```

Task 10:

Find max and min between 3 numbers.

```
MOV AX, 2       ; First number
MOV BX, 3       ; Second number
MOV CX, 4       ; Third number

; Find max
MOV DX, AX      ; Assume AX is max
CMP BX, DX
JLE SKIP1
MOV DX, BX
SKIP1:
CMP CX, DX
JLE SKIP2
MOV DX, CX
SKIP2:

; DX now has max

; Find min
MOV SI, AX      ; Assume AX is min
CMP BX, SI
JGE SKIP3
MOV SI, BX
SKIP3:
CMP CX, SI
JGE SKIP4
MOV SI, CX
SKIP4:

; Now display max and min (you can call two output sections here)
```

Task 11:

Check if the triangle is valid.

A triangle is valid if:
a + b > c, a + c > b, and b + c > a

```
MOV AX, 5       ; a
MOV BX, 6       ; b
MOV CX, 7       ; c

MOV DX, AX
ADD DX, BX
CMP DX, CX
JLE INVALID

MOV DX, AX
ADD DX, CX
CMP DX, BX
JLE INVALID

MOV DX, BX
ADD DX, CX
CMP DX, AX
JLE INVALID

; All conditions passed
MOV DL, 'Y'
JMP PRINT

INVALID:
MOV DL, 'N'

PRINT:
MOV AH, 2
INT 21H
```

Task 12:

Digit mapping:

- 0–3 → 'i'
- 4–6 → 'k'
- 7–9 → 'l'
- 10 → 'm'

```
MOV AL, 7

CMP AL, 4
JL PRINT_I

CMP AL, 7
JL PRINT_K

CMP AL, 10
JL PRINT_L

CMP AL, 10
JE PRINT_M
JMP END

PRINT_I:
MOV DL, 'i'
JMP DISP

PRINT_K:
MOV DL, 'k'
JMP DISP

PRINT_L:
MOV DL, 'l'
JMP DISP

PRINT_M:
```

```
MOV DL, 'm'

DISP:
MOV AH, 2
INT 21H

END:
```

Task 13:

Print day name (1–7), assuming Saturday is day 1.

```
MOV AL, 3

CMP AL, 1
JE SAT
CMP AL, 2
JE SUN
CMP AL, 3
JE MON
CMP AL, 4
JE TUE
CMP AL, 5
JE WED
CMP AL, 6
JE THU
CMP AL, 7
JE FRI
JMP END

SAT: ; Saturday
MOV DX, OFFSET MSG_SAT
JMP PRINT

SUN:
MOV DX, OFFSET MSG_SUN
JMP PRINT

MON:
```

```
MOV DX, OFFSET MSG_MON
JMP PRINT

TUE:
MOV DX, OFFSET MSG_TUE
JMP PRINT

WED:
MOV DX, OFFSET MSG_WED
JMP PRINT

THU:
MOV DX, OFFSET MSG_THU
JMP PRINT

FRI:
MOV DX, OFFSET MSG_FRI
JMP PRINT

PRINT:
MOV AH, 9
INT 21H

END:
INT 20H

MSG_SAT DB 'Saturday$'
MSG_SUN DB 'Sunday$'
MSG_MON DB 'Monday$'
MSG_TUE DB 'Tuesday$'
MSG_WED DB 'Wednesday$'
MSG_THU DB 'Thursday$'
MSG_FRI DB 'Friday$'
```

Task 14:

Print number of days in a month (input = 1 to 12)

```
MOV AL, 3      ; User Input

CMP AL, 1
JE MONTH_31
CMP AL, 3
JE MONTH_31
CMP AL, 5
JE MONTH_31
CMP AL, 7
JE MONTH_31
CMP AL, 8
JE MONTH_31
CMP AL, 10
JE MONTH_31
CMP AL, 12
JE MONTH_31

CMP AL, 4
JE MONTH_30
CMP AL, 6
JE MONTH_30
CMP AL, 9
JE MONTH_30
CMP AL, 11
JE MONTH_30

CMP AL, 2
JE MONTH_28

JMP END

MONTH_31:
MOV DX, OFFSET MSG_31
JMP PRINT

MONTH_30:
MOV DX, OFFSET MSG_30
JMP PRINT
```

```
MONTH_28:
MOV DX, OFFSET MSG_28
JMP PRINT

PRINT:
MOV AH, 9
INT 21H

END:
INT 20H

MSG_31 DB '31 day$', 0
MSG_30 DB '30 day$', 0
MSG_28 DB '28 or 29 day$', 0
```