Task 01:

; Q1 solution starts from here

```
.model small
saveReg macro R1, R2, R3
   push R1
   push R2
   push R3
endm
restoreReg macro S1, S2, S3
   pop S3
   pop S2
   pop S1
endm
displayString macro str
   saveReg DX, AX, BX
   lea DX, str
   mov AH, 9
   int 21h
   restoreReg DX, AX, BX
endm

.data
msg db 'ROY$'
.stack 3000h
.code
MAIN PROC
   mov ax, @data
   mov ds, ax
   mov es, ax
   displayString msg
   mov ax, 4C00h
   int 21h
MAIN ENDP
END MAIN
```

; Q1 solution ends here

Task 02:

; Q2 solution starts from here

```
.model small
saveReg macro R1, R2, R3
    push R1
    push R2
    push R3
endm
restoreReg macro S1, S2, S3
    pop S1
    pop S2
    pop S3
endm
factorial macro number, result
    mov ax, 1
    mov cx, [number]
factorial_loop:
    cmp cx, 1
    jle factorial_done
    mul cx
    dec cx
    jmp factorial_loop
factorial_done:
    mov [result], ax
endm

.data
num dw 5
fact dw 1
.stack 3000h
.code
MAIN PROC
    mov ax, @data
    mov ds, ax
    mov es, ax
    factorial num, fact
    mov ax, 4C00h
```

```
    int 21h
MAIN ENDP
END MAIN
```

; Q2 solution ends here

Task 03:

; Q3 solution starts from here

```
.model small
saveReg macro R1, R2, R3
    push R1
    push R2
    push R3
endm
restoreReg macro S1, S2, S3
    pop S1
    pop S2
    pop S3
endm
reverse macro source, length
    saveReg CX, SI, DI
    lea SI, source
    lea DI, source
    add DI, length
    dec DI
    CLD
    mov CX, length
    shr CX, 1
reverse_loop:
    mov AL, [SI]
    mov BL, [DI]
    mov [SI], BL
    mov [DI], AL
    inc SI
    dec DI
```

```
    loop reverse_loop
    restoreReg DI, SI, CX
endm

.data
str1 db 'ABCD$', 0
length1 dw 4
.stack 3000h
.code
MAIN PROC
    mov ax, @data
    mov ds, ax
    mov es, ax
    reverse str1, length1

    mov ah, 9
    lea DX, str1
    int 21h
    mov ax, 4c00h
    int 21h
MAIN ENDP
END MAIN
```

; Q3 solution ends here

Task 04:

Q4; solution starts from here

```
.model small
saveReg macro R1, R2, R3
    push R1
    push R2
    push R3
```

```
endm
restoreReg macro S1, S2, S3
    pop S1
    pop S2
    pop S3
endm

.data
equation db "(A+B)*(C-D)$"
openCount dw 0
closeCount dw 0
result db "PARENTHESIS BALANCED$"
notBalanced db "PARENTHESIS NOT BALANCED$"
.stack 3000h
.code
MAIN PROC
    mov ax, @data
    mov ds, ax
    mov es, ax
    lea SI, equation
    mov CX, 0

check_loop:
    mov AL, [SI]
    cmp AL, '$'
    je check_result
    cmp AL, '('
    jne check_close
    inc openCount
    jmp next_char
check_close:
    cmp AL, ')'
    jne next_char
    inc closeCount
    mov AX, closeCount
    cmp AX, openCount
    jg not_balanced
next_char:
    inc SI
    jmp check_loop
check_result:
    mov AX, openCount
    cmp AX, closeCount
    jne not_balanced
```

```
    lea DX, result
    jmp print_result
not_balanced:
    lea DX, notBalanced
print_result:
    mov ah, 9
    int 21h
    mov ax, 4c00h
    int 21h
MAIN ENDP
END MAIN
```

Q4; solution ends here

Task 05:

```
;  solution starts from here
.model small
saveReg macro R1, R2, R3
    push R1
    push R2
    push R3
endm
restoreReg macro S1, S2, S3
    pop S1
    pop S2
    pop S3
endm
findMax macro num1, num2, num3, result
    mov AX, [num1]
    cmp AX, [num2]
    jge check_third
    mov AX, [num2]
check_third:
    cmp AX, [num3]
    jge done
    mov AX, [num3]
done:
    mov [result], AX
endm
```

```
.data
num1 dw 25
num2 dw 48
num3 dw 30
maxNum dw ?
.stack 3000h
.code
MAIN PROC
    mov ax, @data
    mov ds, ax
    mov es, ax
    findMax num1, num2, num3, maxNum
    mov ax, [maxNum]
    mov ax, 4c00h
    int 21h

MAIN ENDP
END MAIN
```

; solution ends here

Task 06:

```
;  solution starts from here
org 100h
.model small
.stack 100h
.data
numList db 6, 3, 5, 1
n dw 4
maxResult db 0
.code
main proc
    mov ax, @data
    mov ds, ax
    mov cx, n
    lea si, numList
    mov al, [si]
```

```
    mov maxResult, al
    dec cx
    inc si
    cmp cx, 0
    je endMain
find_max:
    mov al, maxResult
    mov bl, [si]
    call max_two
    mov maxResult, al
    inc si
    loop find_max
    call print
endMain:
    mov ah, 4Ch
    int 21h
main endp

max_two proc
    cmp al, bl
    jae end_max_two
    mov al, bl
end_max_two:
    ret
max_two endp

print proc
    mov al, maxResult
    add al, 30h
    mov dl, al
    mov ah, 2
    int 21h
    ret
print endp
end main
```

; solution ends here

Task 07:

```
;  solution starts from here

org 100h
.model small
.stack 100h
.data
x db 5
exponent dw 2
result dw ?
digitCount db 0
.code
main proc
    mov ax, @data
    mov ds, ax
    mov al, x
    mov bl, al

    mov ax, exponent
    mov cx, ax
    call xn
    mov result, ax
    call print
    mov ax, 4c00h
    int 21h
main endp

xn proc
    mov ax, 1
loop1:
    mul bl
    loop loop1
    ret
xn endp

print proc
    mov ax, result
    mov cx, 10
    mov digitCount, 0
    cmp ax, 0
    jne print_loop
    mov dl, '0'
    mov ah, 2
    int 21h
    jmp end_print
```

```
print_loop:
    mov dx, 0
    div cx
    push dx
    inc digitCount
    cmp ax, 0
    jne print_loop
print_digits:
    cmp digitCount, 0
    je end_print
    pop dx
    add dl, '0'
    mov ah, 2
    int 21h
    dec digitCount
    jmp print_digits
end_print:
    ret
print endp
end main


; solution ends here
```

Task 08:

```
;  solution starts from here



org 100h
.model small
.stack 100h
.data
prompt_msg db 'enter a number (1-9): $'
prime_msg db 0ah, 0dh, 'prime numbers are: $'
input_num db 0
newLine db 0ah, 0dh, '$'
.code
main proc
```

```
    mov ax, @data
    mov ds, ax
    mov ah, 09h
    lea dx, prompt_msg
    int 21h
    mov ah, 01h
    int 21h
    sub al, '0'
    mov input_num, al
    mov ah, 09h
    lea dx, prime_msg
    int 21h
    call printprimes

    mov ah, 09h
    lea dx, newLine
    int 21h
    mov ah, 4ch
    int 21h
main endp

printprimes proc
    mov al, input_num
    mov cx, ax
    mov bx, 2
check_next:
    cmp bx, cx
    jae done_prime_check
    mov di, bx
    mov si, 2
is_prime:
    cmp si, di
    jge prime_found
    mov ax, di
    xor dx, dx
    div si
    cmp dx, 0
    je not_prime
    inc si
    jmp is_prime
prime_found:
    mov ax, bx
    add al, '0'
    mov dl, al
```

```
    mov ah, 02h
    int 21h
    mov dl, ' '
    mov ah, 02h
    int 21h
    jmp continue_check
not_prime:
    jmp continue_check

continue_check:
    inc bx
    jmp check_next
done_prime_check:
    ret
printprimes endp
end main

; solution ends here
```

Task 09:

```
;  solution starts from here
org 100h
.model small
.stack 100h
.data
array db 5, 1, 7, 2, 9, 4
found db 0
name1 db "found$"
name2 db "not found$"
.code
main proc
    mov ax, @data
    mov ds, ax
    mov ah, 1
    int 21h
    sub al, 30h
    call search
    call print
```

```
    mov ax, 4c00h
    int 21h
main endp

search proc
    mov si, 0
    mov cx, 6
loop1:
    mov bh, array[si]

    cmp al, bh
    je value_found
    inc si
    loop loop1
    jmp search_end
value_found:
    mov found, 1
search_end:
    ret
search endp

print proc
    cmp found, 1
    je print1
print2:
    lea dx, name2
    mov ah, 9
    int 21h
    jmp print_end
print1:
    lea dx, name1
    mov ah, 9
    int 21h
print_end:
    ret
print endp
end main



; solution ends here
```