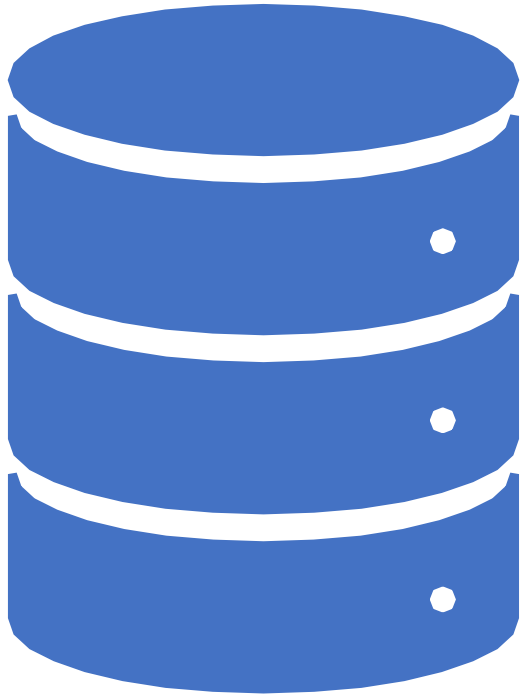


ASSEMBLY LANGUAGE

STACK

Stack. Use

1) Undo/Redo. Options. (Used in MS word, Excel, Notepad etc).
2). Back Foreward.
3). Solving Mathematical Problems with precedence.

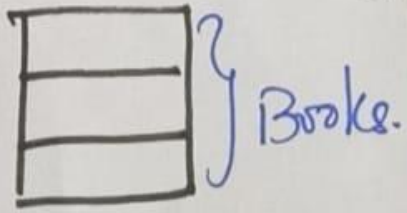
Use of stack in CS.

Stack in Assembly language. Use

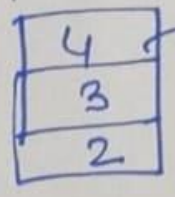
1) Swap two Numbers.
2) To Reverse a string
3) Helps in Nested Loops (loop within Loop).

Stack:- Arrangement, Organization.

LIFO (last In First Out):



In assembly we reserve part of RAM., Insert values in it.



→ Pick first.

"Stack is a DS (data structure) that works on LIFO principle."

→ Stack is a name for data arrangement. like LIFO.

USE OF STACK IN ASSEMBLY PROGRAM.

→ .stack 100h ⟶ Directive

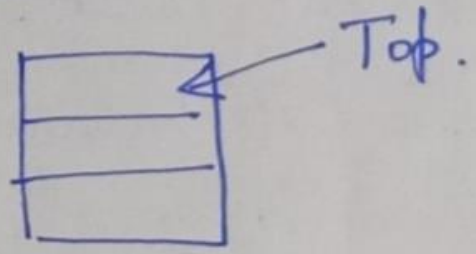
100 Bytes | of memory reserved.

↳ Its a directive/command that reserve 100 bytes for stack.

→ .stack 100h , written after .model small

~~address does to be Release Value from stack?~~

In assembly, we reserve space of RAM, who tell assembly program that which location of RAM is reserved & where is Top?



→ First we reserve space in .stack 100h

→ Space from where to where.

Stack Segment Register hold address of that portion of RAM.

Stack pointer Register points to Top

Stack Segment Register :- Hold address of space reserved for stack.

Stack pointer Register :- Point the top of space reserved for stack.

We write

SS : SP

Add Value to Stack & Take Out-

2 functions Used.. 1) Push
2) Pop.

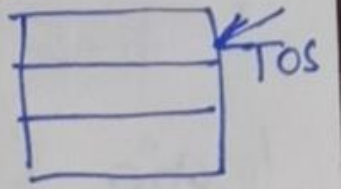
PUSH Register / Variable.

PUSH AX OR PUSH Var1

Value placed in Register / Variable go to stack.

PUSH :- Copies Content from operand to top of stack.

PUSH Reg/Var.



POP Register/Variable.

POP Ax OR POP Var1.

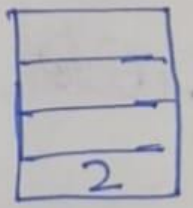
POP :- Copies content from TOS (top of stack) to operand

$ Remember :- the register in PUSH, must be ≥ 16 bits (not DByte., not 8bits)

⇒ Variable also must be 16 bits. (DW)

```
Push al
Pop al  x
```

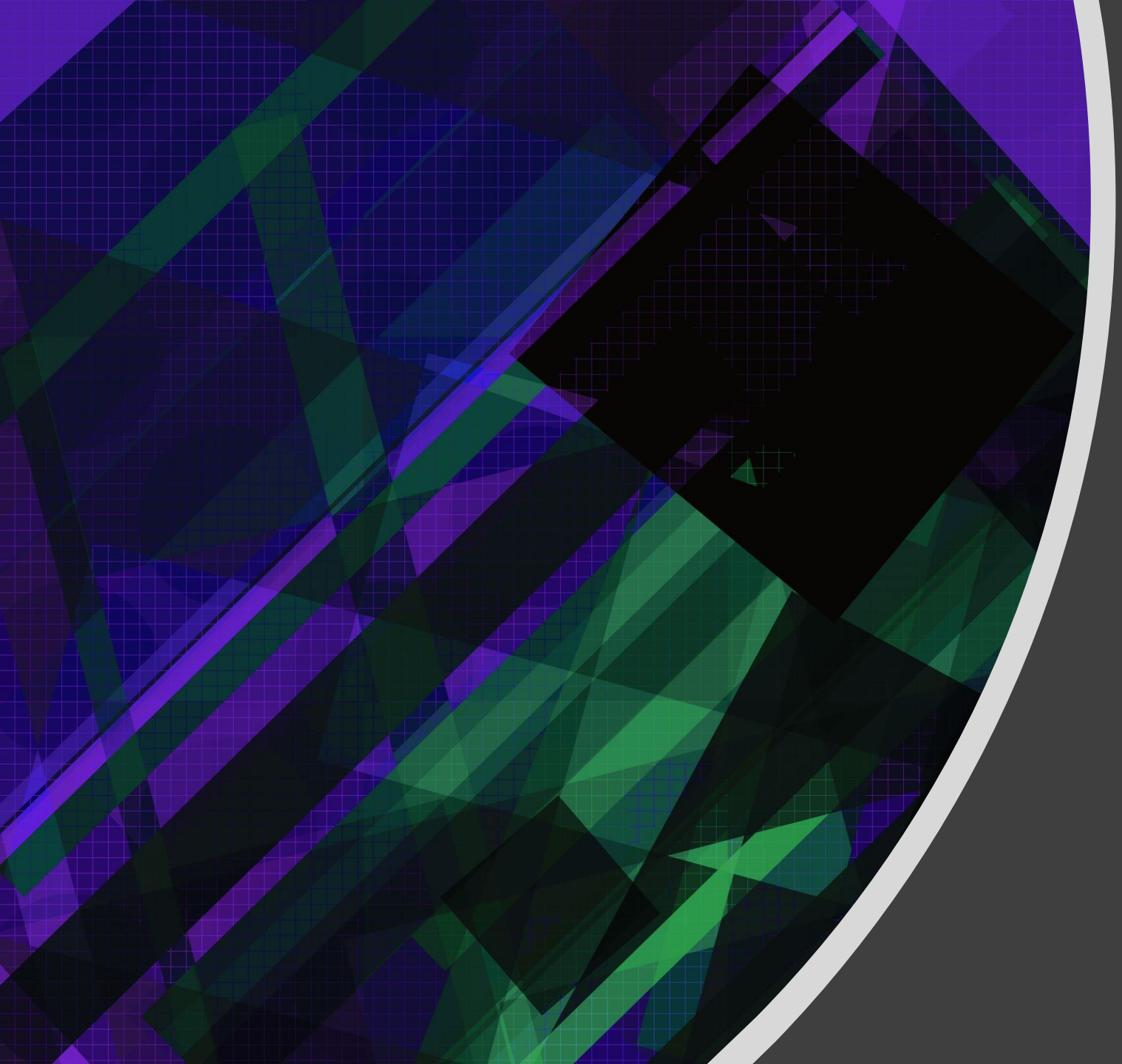
Program Example of Push and Pop.



- model small
- stack 100h
- data
- code

Main Proc

```
Mov AX, 2      ⟶  2 goes to AX.
PUSH AX        ⟶  2 go to stack
POP AX         ⟶  2 again go to AX.
mov DX, AX
mov ah, 2
int 21h
mov ah, 4ch
int 21h
main endp
end main
```



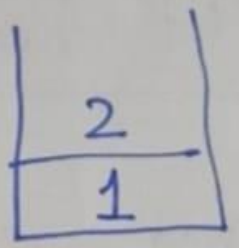
STACK LAB

Program To Swap Two Numbers.

If a = 3, b = 5 After Swap a = 5, b = 3

Logic Swap using help of push and pop.

```
mov ax,'1'
Push ax  ——→ value 1 go to stack.
mov bx,'2'
Push bx  ——→ value 2 go to stack
```



Top is 2, Pop operation 2 is retrieved, (LIFO).

For swapping we write

```
Pop ax ; moves 2 from stack to ax
Pop bx ; moves 1 from stack to bx.
```

Code.

```
; Then print both
mov dx, ax
mov ah, 2
int 21h
```

Repeat same with bx

End Program.

Program to Reverse a String.

Logic :-

1). .data ⟶ string db 'abc' ; create string

2). .code
main proc
mov ax, @data
mov ds, ax

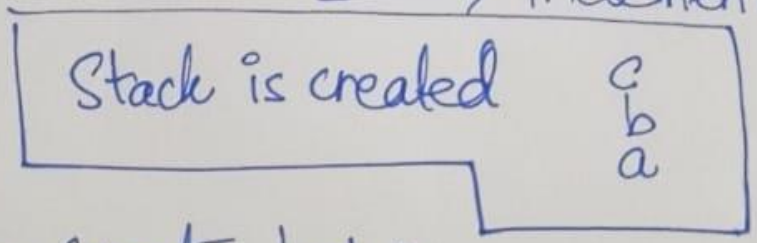
3). To access string character wise

mov si, offset string ⟶ (we send to si, offset and variable (first element address of var. go to si) means starting address.

then we put character one by one into stack then pop, With the help of push and pop, 3 characters, 3 times loop to Run a/c to string.

```
mov cx,3
```
→ loop run 3 times.

```
L1:
  mov bx,[si]
  push bx
  inc si
  loop L1
```
→ starting address value mov to bx.
→ value go to stack.
→ increment.



For pop and print we create loop2

```
mov cx,3
L2:
  pop dx
  mov ah,2
  int 21h
loop L2
```
→ for print place value direct to dx.

```
mov ah, 4ch
int 21h
main endp
end main
```