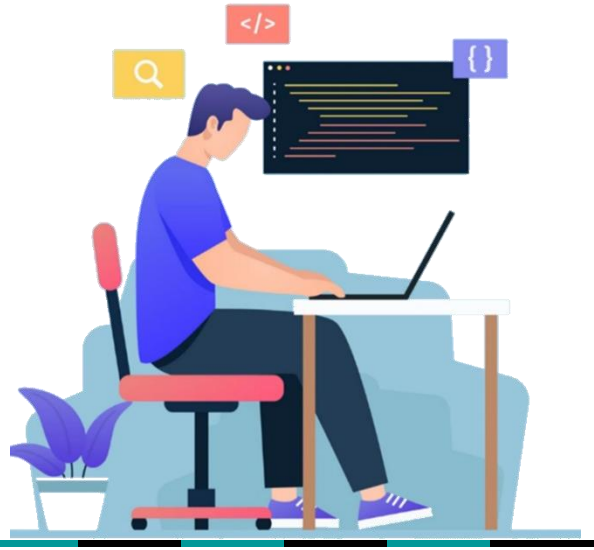
اصول و مبانی برنامه‌نویسی



مجید شبیری

کارشناسی ارشد IT، گرایش شبکه

از دانشگاه صنعتی امیرکبیر

تبدیل الگوریتم به کد

Simple

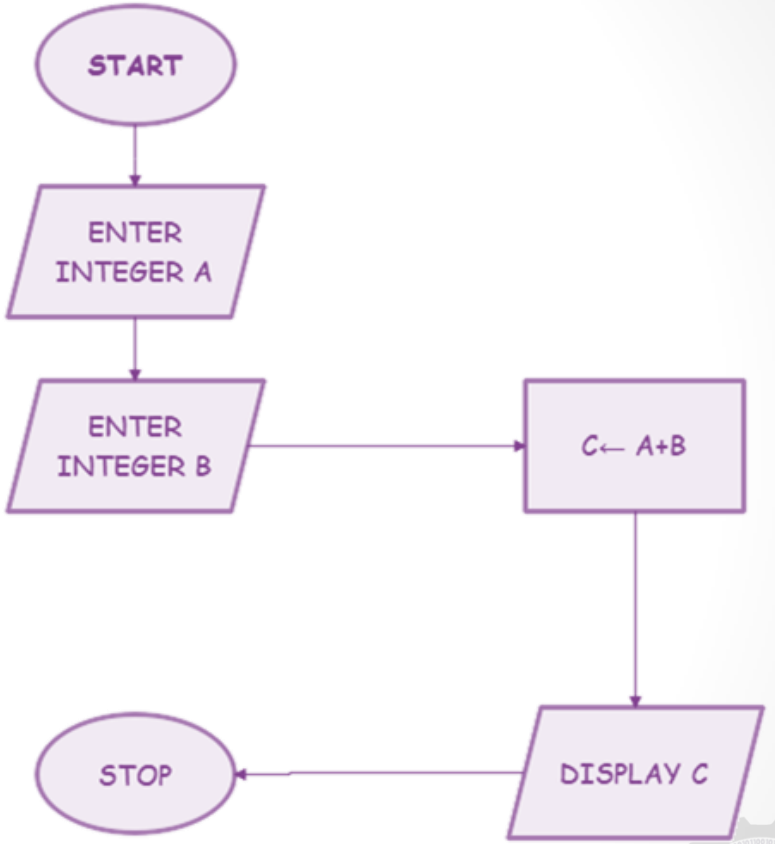
۱- الگوریتم جمع دو عدد

1- Read the Integer **A**.

2- Read Integer **B**.

3- Perform the addition by using the formula: **C= A + B.**

4- Print the Integer **C**.



۲- الگوریتم جابجایی مقدار دو متغیر (swap)

```
START
   Var1, Var2, Temp
   Step 1 → Copy value of Var1 to Temp
   Step 2 → Copy value of Var2 to Var1
   Step 3 → Copy value of Temp to Var2
STOP
```

```
procedure swap(a, b)

   set temp to 0
   temp ← a
   a ← b       // a holds value of b
   b ← temp    // b holds value of a stored in temp

end procedure
```

۳- الگوریتم جابجایی مقدار دو متغیر (swap) – بدون استفاده از متغیر کمکی

```
START
   Var1, Var2
   Step 1 → Add Var1 and Var2 and store to Var1
   Step 2 → Subtract Var2 from Var1 and store to Var2
   Step 3 → Subtract Var2 from Var1 and store to Var1
STOP
```

```
procedure swap(a, b)

   a ← a + b    // a holds the sum of both
   b ← a - b    // b now holds the value of a
   a ← a - b    // a now holds value of b

end procedure
```

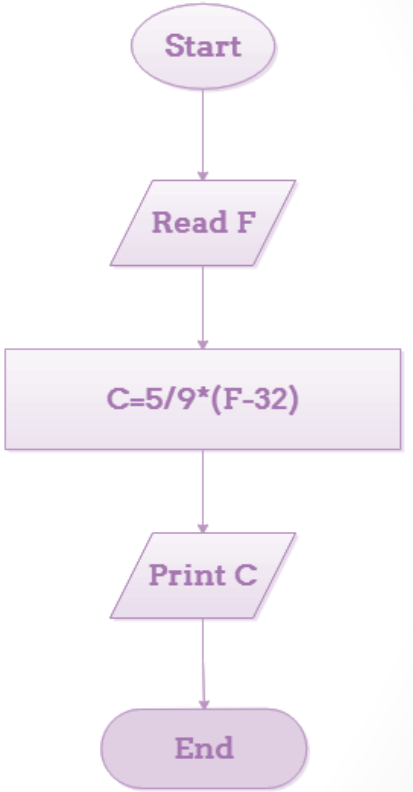
٤- الگوریتم تبدیل دمای فارنهایت به سلسیوس

1- Read temperature in **Fahrenheit** (**F**),

2- Calculate temperature with formula :

$$C = 5/9 * (F - 32)$$

3- Print **C**.



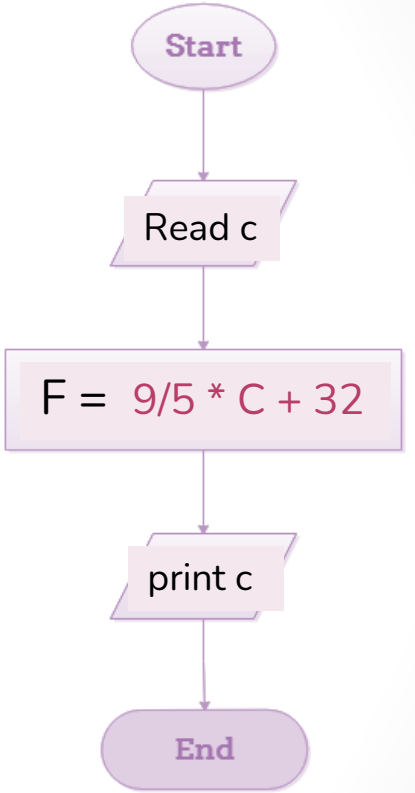
۵- الگوریتم تبدیل دمای سلسیوس به فارنهایت

1- Read temperature in **Fahrenheit** (**F**),

2- Calculate temperature with formula :

F = 9/5 * C + 32

3- Print **F**.



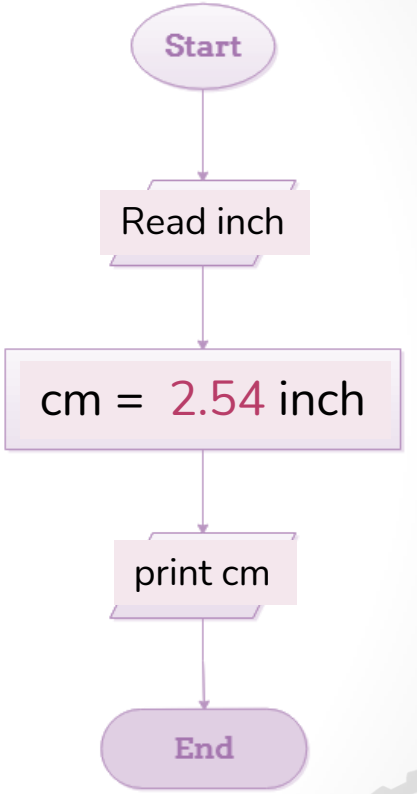
٦- الگوریتم تبدیل اینچ به سانتیمتر

1- Read size in **inch**,

2- Calculate cm with formula :

cm = 2.54 inch

3- Print **cm**.



۷- الگوریتم محاسبه درصد یک مقدار نسبت به کل

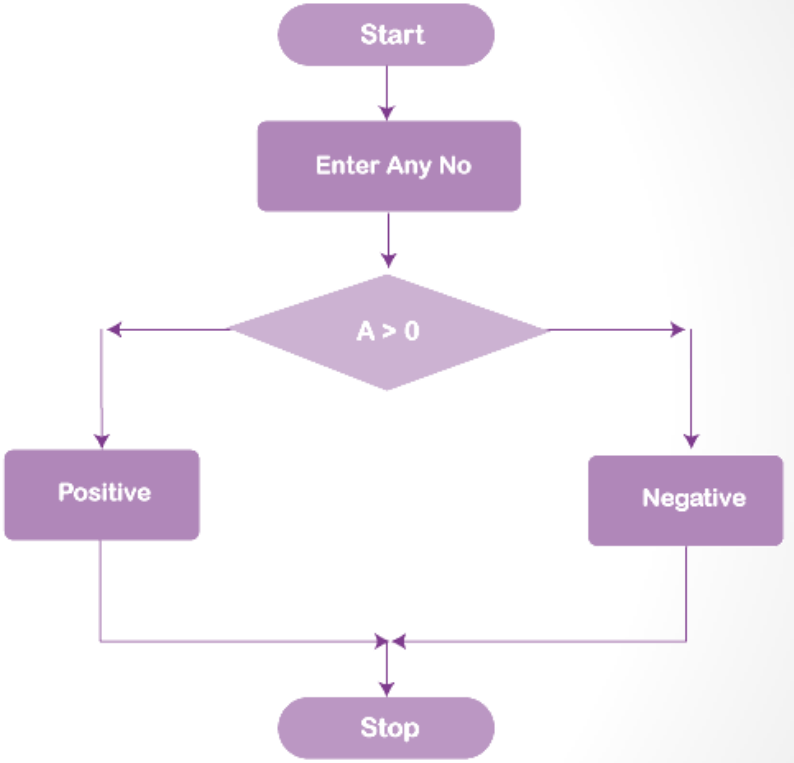
$$percentage = (part / total) \times 100$$

```
START
   Step 1 → Collect values for part and total
   Step 2 → Apply formula { percentage = ( part / total ) × 100 }
   Step 3 → Display percentage
STOP
```

```
procedure percentage()

   VAR value, total
   percentage = value / total * 100
   RETURN percentage

end procedure
```

۸- الگوریتم تشخیص مثبت و منفی

```
START
   Step 1 → Take integer variable A
   Step 2 → Assign value to the variable
   Step 3 → Check if A is greater than or equal to 0
   Step 4 → If true print A is positive
   Step 5 → If false print A is negative
STOP
```




```
procedure positive_negative()

   IF ( number ≥ 0 )
      PRINT number is positive
   ELSE
      PRINT number is negative
   END IF

end procedure
```

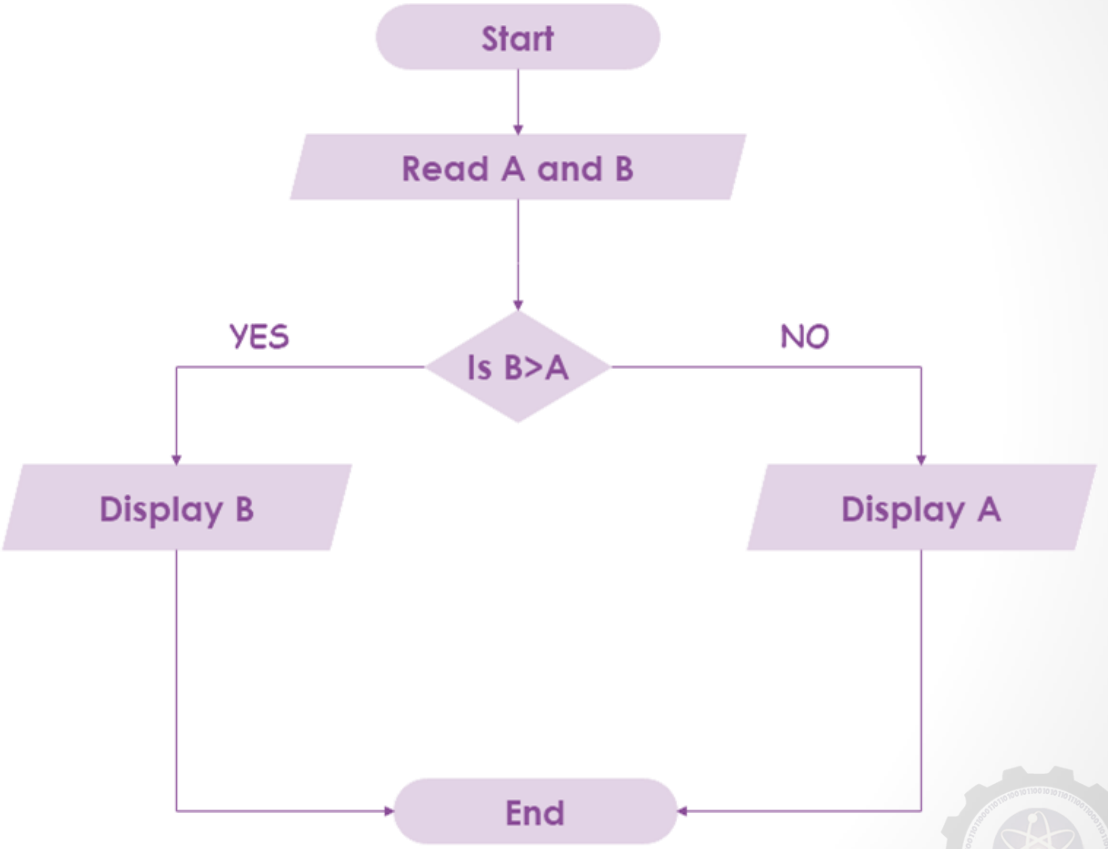
۹- الگوریتم پیداکردن max دو عدد

1- Read the Integer **A**.

2- Read Integer **B**.

3- If **B** is greater than **A**, then print **B**,

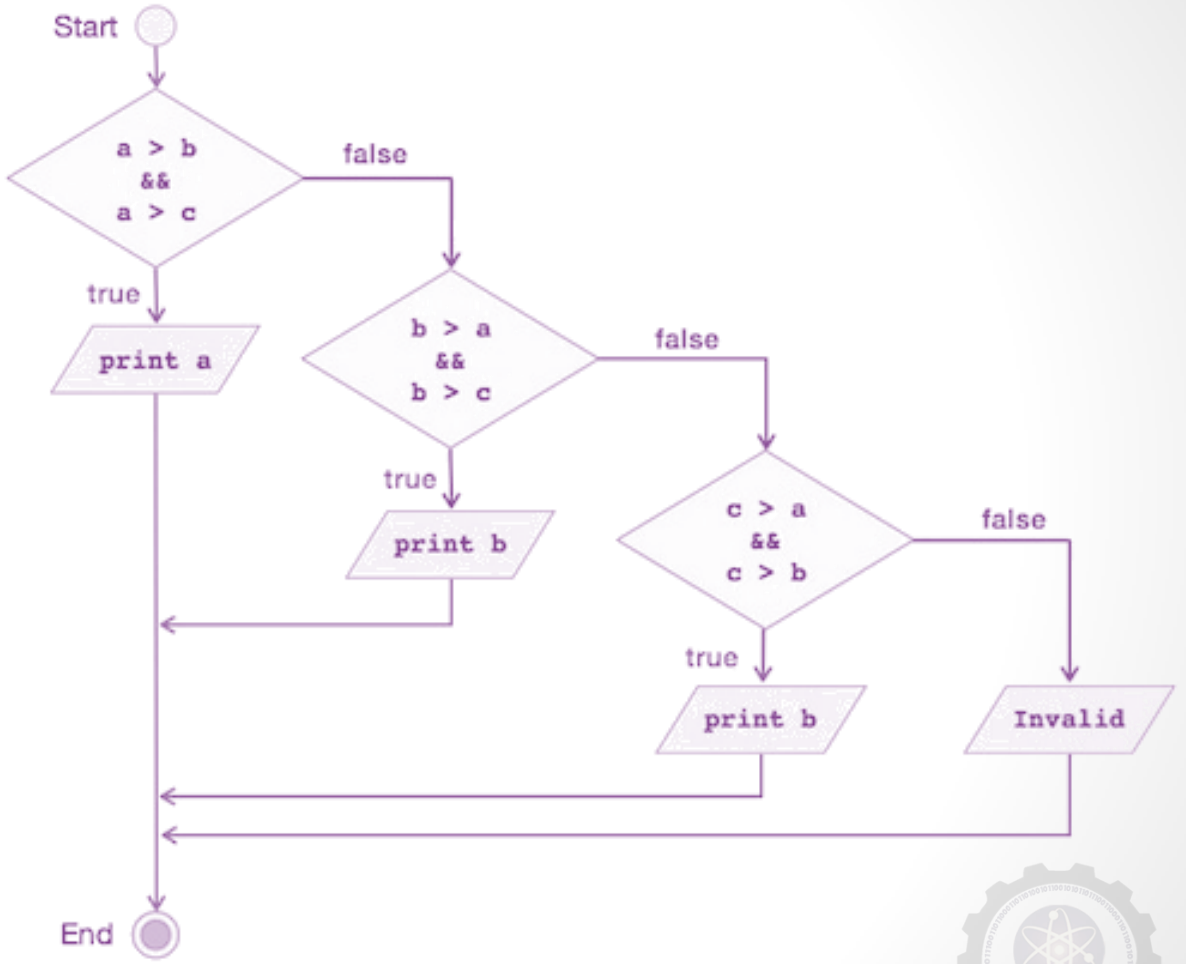
else print **A**.



۱۰- الگوریتم پیداکردن max سه عدد

```
START
   Step 1 → Take two integer variables, say A, B& C
   Step 2 → Assign values to variables
   Step 3 → If A is greater than B & C, Display A is largest value
   Step 4 → If B is greater than A & C, Display B is largest value
   Step 5 → If C is greater than A & B, Display A is largest value
   Step 6 → Otherwise, Display A, B & C are not unique values
STOP
```

```
procedure compare(A, B, C)

   IF A is greater than B AND A is greater than C
      DISPLAY "A is the largest."
   ELSE IF B is greater than A AND A is greater than C
      DISPLAY "B is the largest."
   ELSE IF C is greater than A AND A is greater than B
      DISPLAY "C is the largest."
   ELSE
      DISPLAY "Values not unique."
   END IF

end procedure
```



۱۱- الگوریتم تشخیص زوج و فرد

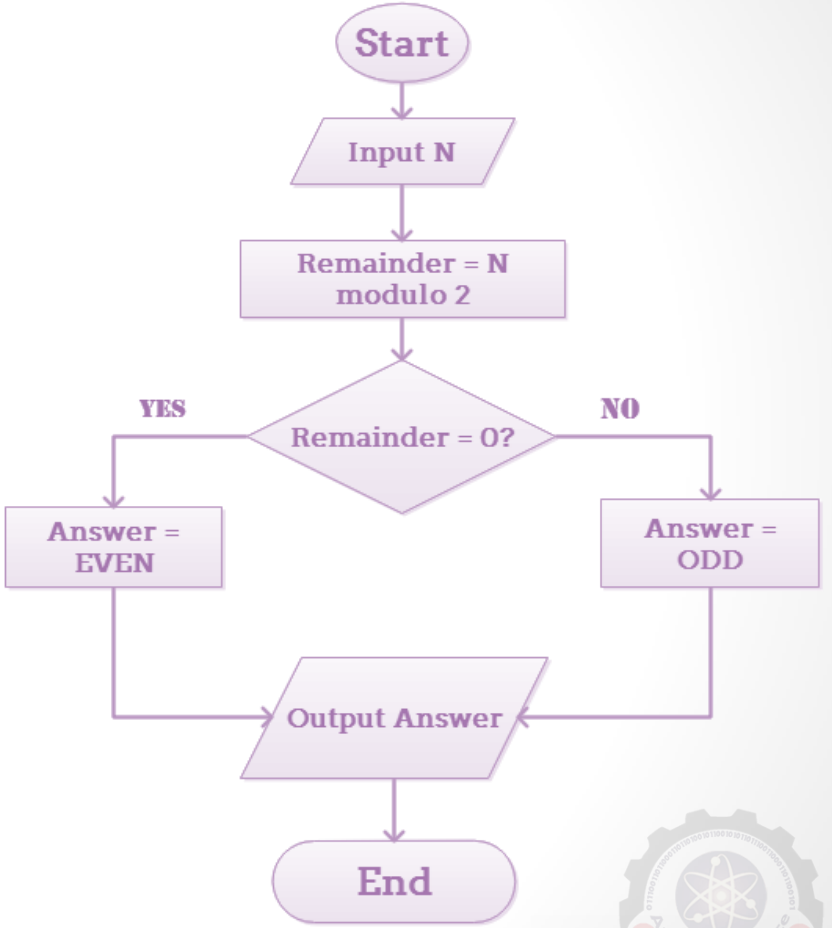
1- Read number **N**.

2- Set **remainder** as **N modulo 2**.

3-**If** the **remainder** is equal to **0** then number **N** is **even**,

else number **N** is **odd**.

4-Print **output**.



۱۲- الگوریتم بزرگترین خارج قسمت مشترک (HCF)

12 → 1, 2, 3, 4, 6, 12

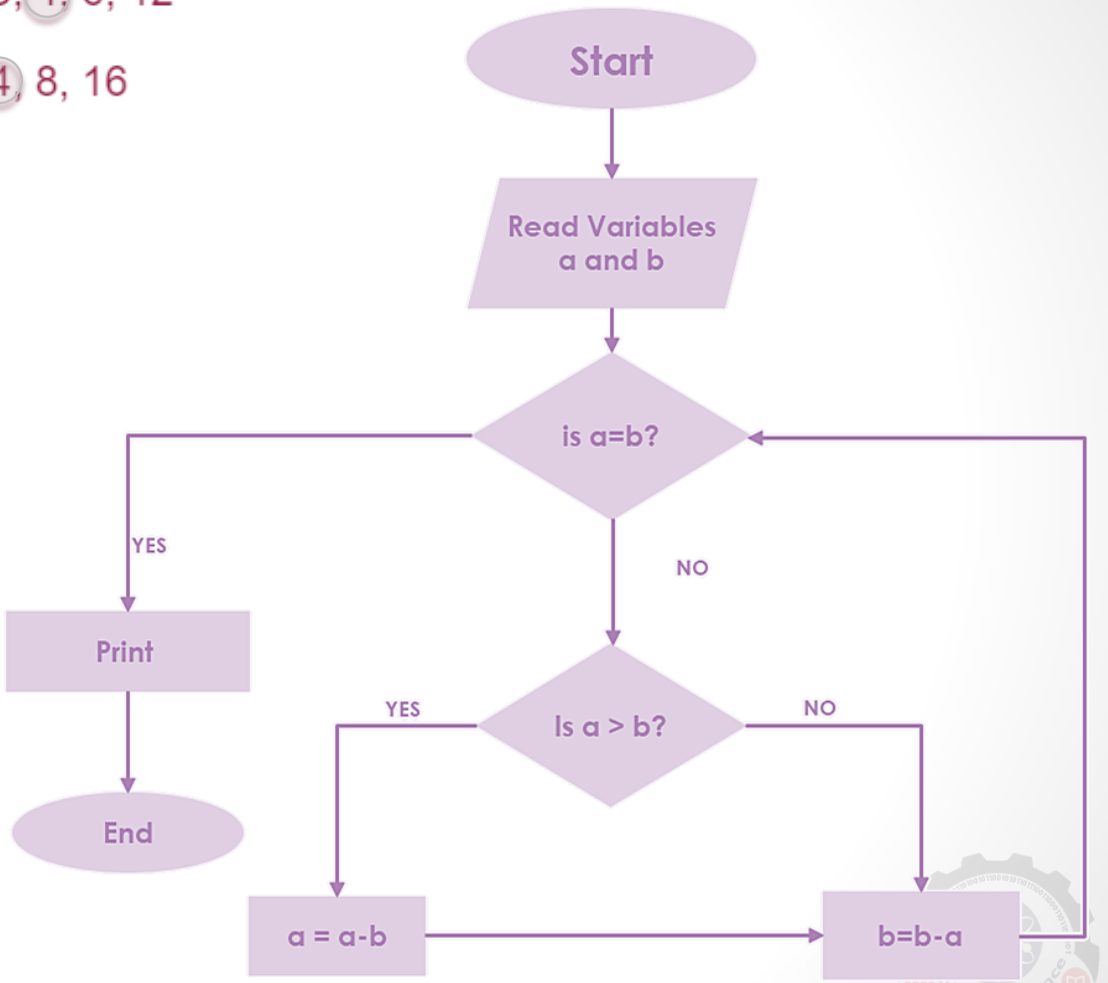
16 → 1, 2, 4, 8, 16

1- Read the variables **a** and **b**.

2- If **a = b**, go to step 4.

3- If **a > b**, then: **a = a - b**. Return to step 2.

4- Print **a or b**



۱۳- الگوریتم کوچکترین مضرب مشترک (LCM)

3 → 3, 6, 9, **12**, 15 ...

4 → 4, 8, **12**, 16, 20 ...

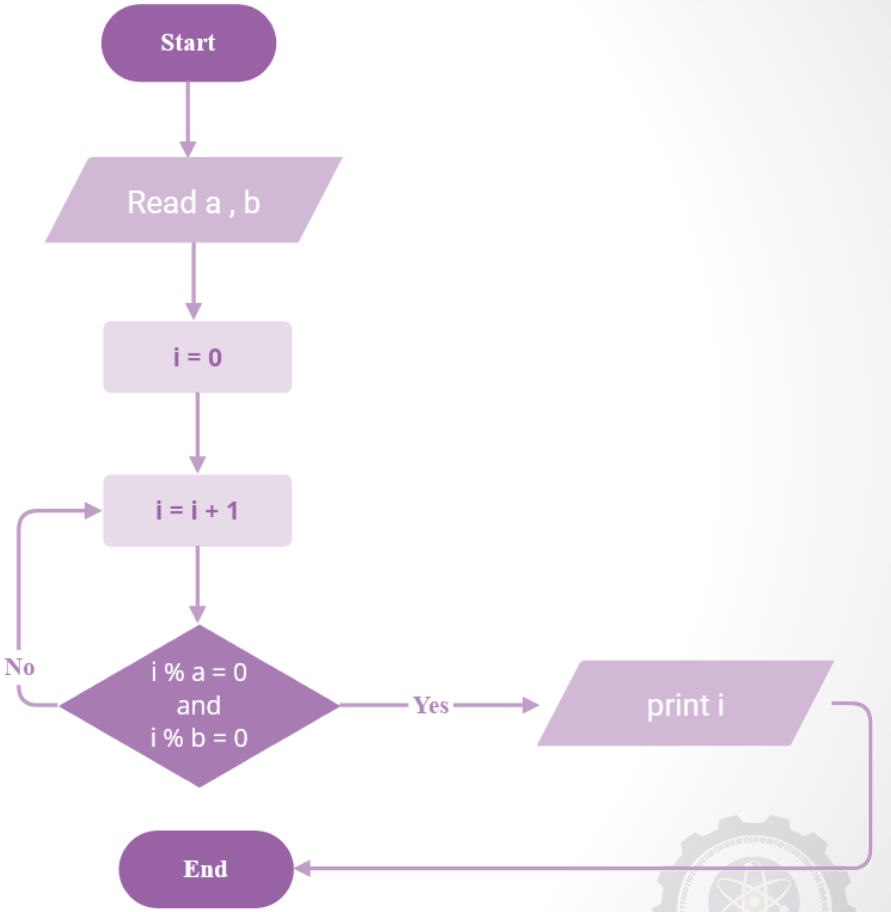
1- Read the variables **a**, **b** and set **i = 0**

2- **i= i +1**

3- If **i % a** is **0** and **i % b** is **0** go to step 4

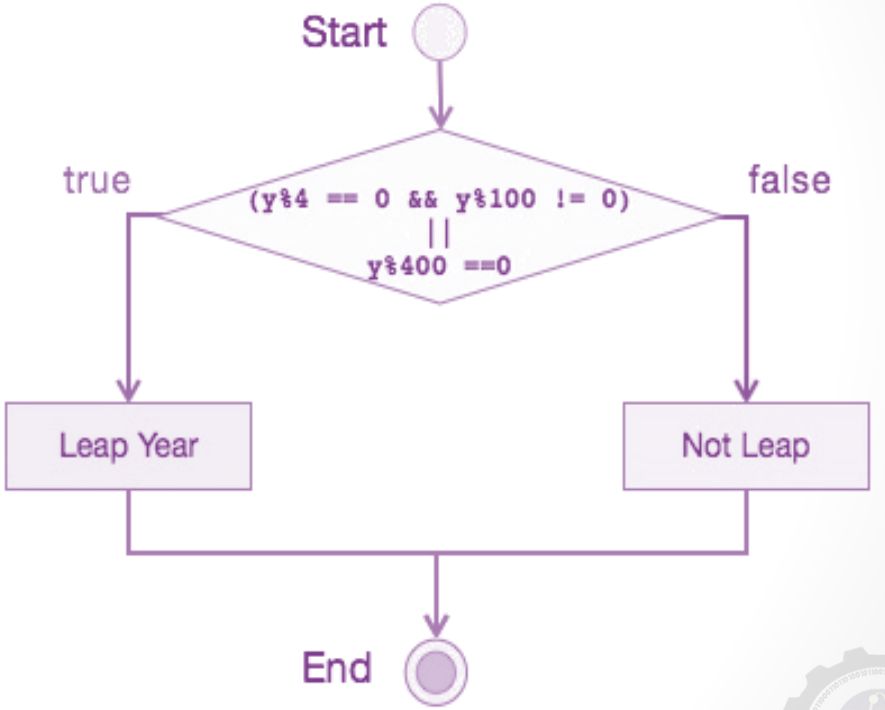
else Return to step 2.

4- Print **i**



۱۴- الگوریتم تشخیص سال کبیسه

```
START
   Step 1 → Take integer variable year
   Step 2 → Assign value to the variable
   Step 3 → Check if year is divisible by 4 but not 100, DISPLAY "leap year"
   Step 4 → Check if year is divisible by 400, DISPLAY "leap year"
   Step 5 → Otherwise, DISPLAY "not leap year"
STOP
```




```
procedure leap_year()

   IF year%4 = 0 AND year%100 != 0 OR year%400 = 0
      PRINT year is leap
   ELSE
      PRINT year is not leap
   END IF

end procedure
```

Loop

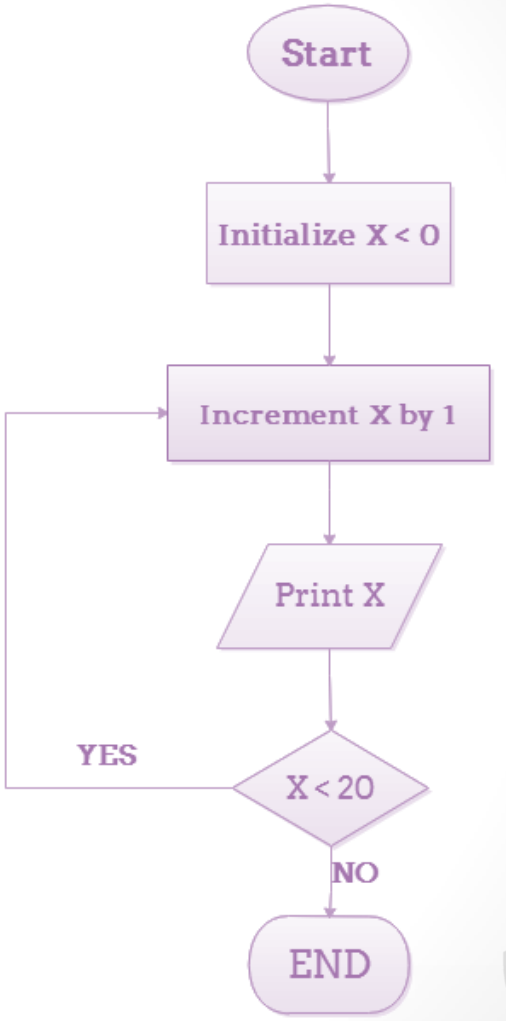
۱۵- الگوریتم پرینت اعداد ۱ تا ۲۰

1- Initialize **X** as 0,

2- Increment **X** by 1,

3- Print **X**,

4- If **X** is less than **20** then go back to step 2.



۱۶- الگوریتم جمع اعداد ۱ تا ۵۰

1- Declare number **N= 0** and **sum= 0**

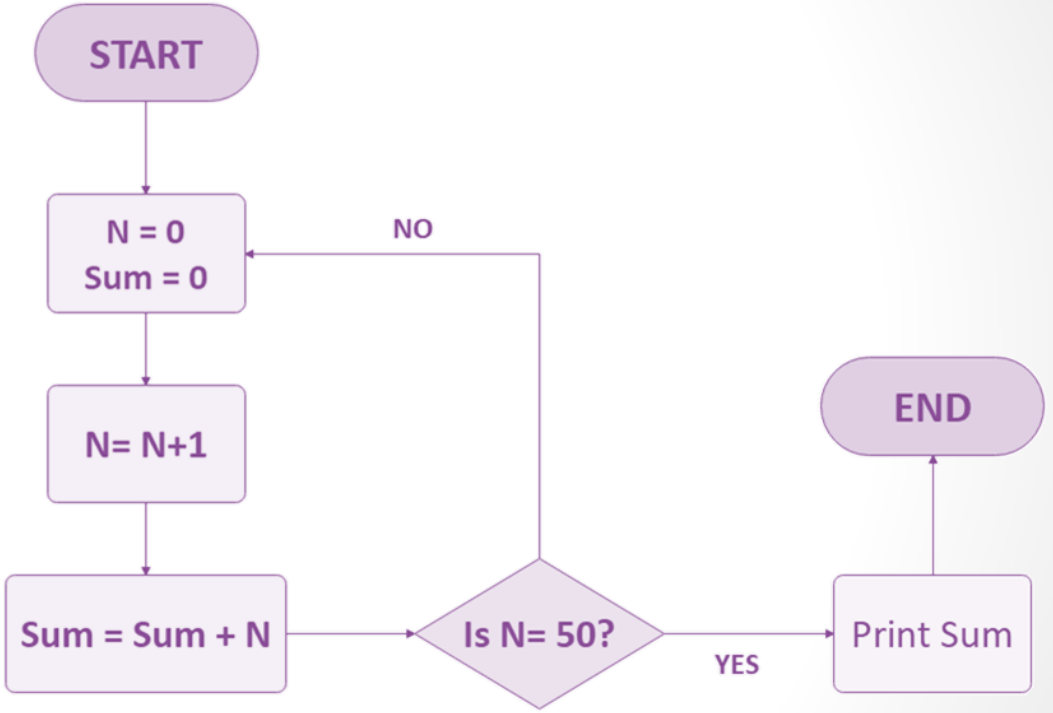
2- Determine N by **N= N+1**

3- Calculate the sum by the formula:

Sum= N + Sum.

4- Add a loop between steps 2 and 3 until **N= 50**.

5- Print **Sum**.



۱۷- الگوریتم شمارش معکوس

```
START
   Step 1 → Define start and end of counting
   Step 2 → Iterate from end to start
   Step 3 → Display loop value at each iteration
STOP
```

```
procedure counting()

   FOR value = END to START DO
      DISPLAY value
   END FOR

end procedure
```

۱۸- الگوریتم جدول شمارش

```
START
   Step 1 → Define Start and End variables
   Step 2 → Outer loop for i from start to end
   Step 3 → Set count to i
   Step 4 → Inner loop for j from 1 to 10
   Step 5 → DISPLAY j * count
   Step 6 → Close inner loop
   Step 7 → Close Outer loop
STOP
```

```
procedure table_of_tables()

   Define start, end
   FOR i = start TO end DO
      count = i
      FOR j = 1 TO 10 DO
         DISPLAY count * j
      END FOR
   END FOR

end procedure
```

۱۹- الگوریتم پیداکردن اعداد زوج

```
START
   Step 1 → Iterate value from 1 to 10
   Step 2 → Check if value is divisible by 2
   Step 3 → If true then display value
STOP
```

```
procedure even_printing(A, B)

   FOR value 1 to 10 DO
      IF value%2 EQUAL TO 0 THEN
         DISPLAY value as even
      END IF
   END FOR

end procedure
```

۲۰- الگوریتم پیداکردن اعداد فرد

```
START
   Step 1 → Iterate value from 1 to 10
   Step 2 → Check if value is divisible by 2
   Step 3 → If false then display value
STOP
```

```
procedure odd_printing(A, B)

   FOR value 1 to 10 DO
      IF value%2 NOT EQUAL TO 0 THEN
         DISPLAY value as even
      END IF
   END FOR

end procedure
```

۲۱- الگوریتم تشخیص عدد اول

```
START
   Step 1 → Take integer variable A
   Step 2 → Divide the variable A with (A-1 to 2)
   Step 3 → If A is divisible by any value (A-1 to 2) it is not prime
   Step 4 → Else it is prime
STOP
```

```
procedure prime_number : number

   FOR loop = 2 to number - 1
      check if number is divisible by loop

      IF divisible
         RETURN "NOT PRIME"
      END IF

   END FOR

   RETURN "PRIME"

end procedure
```

۲۲- الگوریتم تشخیص عدد آرمسترانگ

$$153 = (1)^3 + (5)^3 + (3)^3$$
$$153 = 1 + 125 + 27$$
$$153 = 153$$

START
- Step 1 → Take integer variable Arms
- Step 2 → Assign value to the variable
- Step 3 → Split all digits of Arms
- Step 4 → Find cube-value of each digits
- Step 5 → Add all cube-values together
- Step 6 → Save the output to Sum variable
- Step 7 → If Sum equals to Arms print *Armstrong Number*
- Step 8 → If Sum not equals to Arms print *Not Armstrong Number*

STOP

```
procedure armstrong : number

   check = number
   rem   = 0

   WHILE check IS NOT 0
      rem ← check modulo 10
      sum ← sum + (rem)^3
      divide check by 10
   END WHILE

   IF sum equals to number
      PRINT armstrong
   ELSE
      PRINT not an armstrong
   END IF

end procedure
```

۲۳- چاپ مثلث متساوی الاضلاع

Step 1 - Take number of rows to be printed, n.
Step 2 - Make an iteration for n times
Step 3 - Print " " (space) for in decreasing order from 1 to n-1
Step 4 - Print "* " (start, space) in increasing order
Step 5 - Return

```
procedure equi_triangle

   FOR I = 1 to N DO
      FOR J = 1 to N DO
         PRINT " "
      END FOR

      FOR J = 1 to I DO
         PRINT "* "
      END FOR
   END FOR

end procedure
```

۲۴- الگوریتم چاپ مثلث راست گوشه

```
*
* *
* * *
* * * *
* * * * *
```

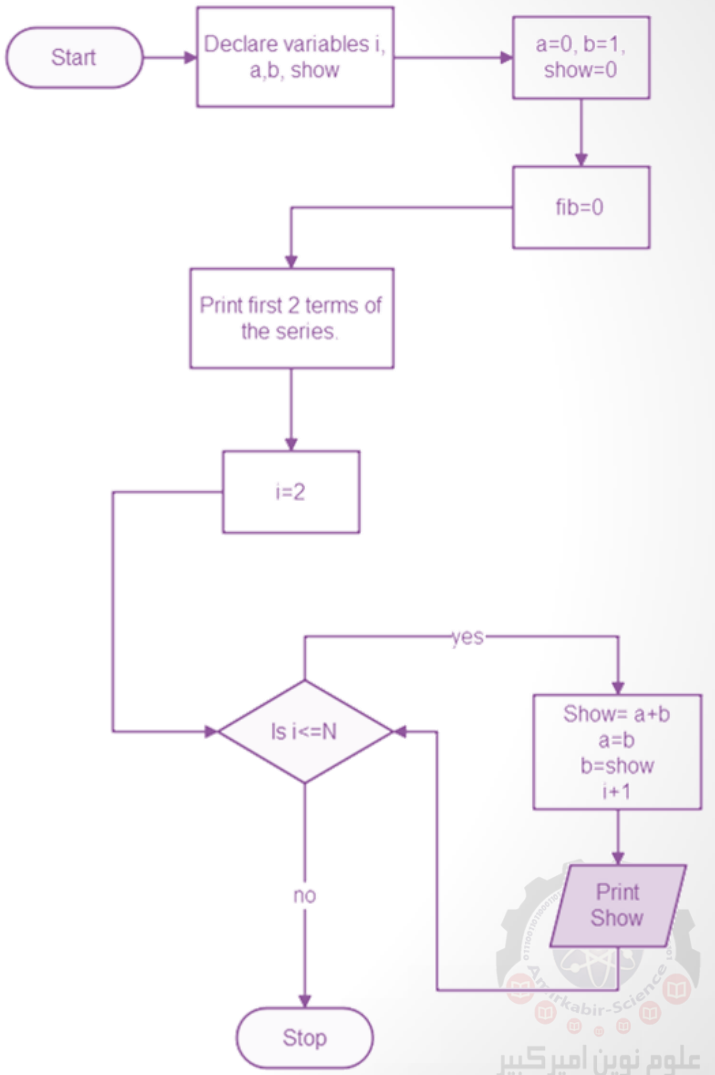
```
Step 1 - Take number of rows to be printed, n.
Step 2 - Make outer iteration I for n times to print rows
Step 3 - Make inner iteration for J to I
Step 3 - Print "*" (star)
Step 4 - Print NEWLINE character after each inner iteration
Step 5 - Return
```

```
procedure right_triangle

   FOR I = 1 to N DO
      FOR J = 1 to I DO
         PRINT "*"
      END FOR
      PRINT NEWLINE
   END FOR

end procedure
```

۲۵- الگوریتم محاسبه دنباله فیبوناچی

$$F_n = F_{n-1} + F_{n-2}$$

$F_8 = 0\ 1\ 1\ 2\ 3\ 5\ 8\ 13$



1- Declare the variables **i**, **a**, **b**, **show**.

2- Enter the values for the variables, **a=0**, **b=1**, **show=0**

3- Enter the **terms of the Fibonacci series** to be printed, i.e=, **1000**.

4- Print **the first two terms** of the series.

5- Loop the following steps:

Show = **a** + **b**

a= **b**

b = **show**

Add **1** to the value of **i** each time.

Print **Show**

۲۶- الگوریتم پیدا کردن ریشه دوم (رادیکال با فرجه ۲)

$$\sqrt[2]{24} = 4.898980$$

```
START
   Step 1 → Define value n to find square root of
   Step 2 → Define variable i and set it to 1 (For integer part)
   Step 3 → Define variable p and set it to 0.00001 (For fraction part)
   Step 4 → While i*i is less than n, increment i
   Step 5 → Step 4 should produce the integer part so far
   Step 6 → While i*i is less than n, add p to i
   Step 7 → Now i has the square root value of n
STOP
```

```
procedure square_root( n )

   SET precision TO 0.00001
   FOR i = 1 TO i*i < n DO
      i = i + 1
   END FOR

   FOR i = i - 1 TO i*i < n DO
      i = i + precision
   END FOR
   DISPLAY i AS square root

end procedure
```

۲۷- الگوریتم محاسبه فاکتوریل

```
START
   Step 1 → Take integer variable A
   Step 2 → Assign value to the variable
   Step 3 → From value A upto 1 multiply each digit and store
   Step 4 → the final stored value is factorial of A
STOP
```

```
procedure find_factorial(number)

   FOR value = 1 to number
      factorial = factorial * value
   END FOR
   DISPLAY factorial

end procedure
```

۲۸- الگوریتم جایگشت و ترکیب k از n

ترکیب (combination) **k از n** : انتخاب k عنصر از یک مجموعه n تایی - بدون ترتیب

جایگشت (permutaion) **k از n** : انتخاب k عنصر از یک لیست n تایی - ترتیب مهم است

$${}_nC_r = n! / ((n - r)! \times r!)$$

$${}_nP_r = n! / (n - r)!$$

```
START
   Step 1 → Define values for n and r
   Step 2 → Calculate factorial of n and (n-r)
   Step 3 → Divide factorial(n) by factorial(n-r)
   Step 4 → Display result as permutation
STOP
```

```
procedure permutation()

   Define n and r
   P = factorial(n) / factorial(n-r)
   DISPLAY P

end procedure
```

List

۲۹- الگوریتم چاپ عناصر آرایه

```
0 1 2 3 4 5 6 7 8 9
```

```
START
   Step 1 → Take an array A and define its values
   Step 2 → Loop for each value of A
   Step 3 → Display A[n] where n is the value of current iteration
STOP
```

```
procedure print_array(A)

   FOR EACH value in A DO
      DISPLAY A[n]
   END FOR

end procedure
```

۳۰- الگوریتم محاسبه مجموع عناصر آرایه

```
START
   Step 1 → Take an array A and define its values
   Step 2 → Loop for each value of A
   Step 3 → Add each element to 'sum' variable
   Step 4 → After the loop finishes, display 'sum'
STOP
```

```
procedure sum_array(A)

   Declare sum as integer
   FOR EACH value in A DO
      sum ← sum + A[n]
   END FOR
   Display sum

end procedure
```

۳۱- الگوریتم یافتن عنصر مینیمم آرایه

```
START
   Step 1 → Take an array A and define its values
   Step 2 → Declare smallest as integer
   Step 3 → Set smallest to 0
   Step 4 → Loop for each value of A
   Step 5 → If A[n] < smallest, Assign A[n] to smallest
   Step 6 → After loop finishes, Display smallest as smallest element of array
STOP
```

```
procedure smallest_array(A)

   Declare smallest as integer
   Set smallest to 0
   FOR EACH value in A DO
      IF A[n] is less than smallest THEN
         smallest ← A[n]
      ENDIF
   END FOR
   Display smallest

end procedure
```

۳۲- الگوریتم مرتب سازی اعداد داخل آرایه (Selection Sort)

1- Initialize minimum value(**min_idx**) to location 0.

2- Traverse the array to find the minimum element in the array.

3- **While** traversing

if any element smaller than **min_idx** is found then

swap both the values.

4- Then, increment **min_idx** to point to the **next element**.

5- **Repeat** until the array is sorted.

۳۳- الگوریتم پیدا کردن مد آرایه

0,6,7,2,7 —mode→ 7

```
START
   Step 1 → Take an integer set A of n values
   Step 2 → Count the occurence of each integer value in A
   Step 3 → Display the value with highest occurence
STOP
```

```
procedure mode()

   Array A
   FOR EACH value i in A DO
      Set Count to 0
      FOR j FROM 0 to i DO
         IF A[i] = A[j]
            Increment Count
         END IF
      END FOR

      IF Count > MaxCount
         MaxCount =  Count
         Value    =  A[i]
      END IF
   END FOR
   DISPLAY Value as Mode

end procedure
```

۳۴- الگوریتم کپی آرایه

```
original -> copied
    1         1
    2         2
    3         3
    4         4
    5         5
    6         6
    7         7
    8         8
    9         9
    0         0
```

```
START
   Step 1 → Take two arrays A, B
   Step 2 → Store values in A
   Step 3 → Loop for each value of A
   Step 4 → Copy each index value to B array at the same index location
STOP
```

```
procedure copy_array(A, B)

   SET index to 1
   FOR EACH value in A DO
      B[index] = A[index]
      INCREMENT index
   END FOR

end procedure
```

۳۵- الگوریتم جداسازی عناصر زوج و فرد آرایه

```
original ->  0 1 2 3 4 5 6 7 8 9
even ->  0 2 4 6 8
odd ->  1 3 5 7 9
```

```
START
   Step 1 → Take three array variables A, E, and O
   Step 2 → Store continuous values in A
   Step 3 → Loop for each value of A
   Step 4 → If A[n] is even then store in E array
   Step 5 → If A[n] is odd then store in E array
STOP
```

```
procedure divide_array(A)

   FOR EACH value in A DO
      IF A[n] is even
         save in E
      ELSE
         save in O
      END IF
   END FOR

end procedure
```

۳۶- الگوریتم ادغام دو آرایه

```
Even ->  0 2 4 6 8
Odd  ->  1 3 5 7 9
Concat ->  0 2 4 6 8 1 3 5 7 9
```

```
START
   Step 1 → Take three array variables A, E, and O
   Step 2 → Store even values in array E
   Step 3 → Store odd values in array O
   Step 4 → Start loop from 0 to sizeof(E)
   Step 5 → Copy E[n] to A[index]
   Step 6 → Start loop from 0 to sizeof(O)
   Step 7 → Copy E[n] to A[index]
   Step 8 → Display A
STOP
```

```
procedure concate_array(A)

   Array E, O
   index ← 0
   FOR EACH value in E DO
      A[index] ← E[n]
      INCREMENT index
   END FOR

   FOR EACH value in O DO
      A[index] ← O[n]
      INCREMENT index
   END FOR
   DISPLAY A

end procedure
```

string

۳۷- چاپ یک string

```
Hello World
```

۳۸- چاپ کاراکترهای یک string

```
H
e
l
l
o

W
o
r
l
d
```

۳۹- محاسبه طول یک string (بدون استفاده از تابع strlen())

```
Length of string 'TajMahal' is 8
```

۴۰- محاسبه تعداد کاراکتر ‘a’ در string

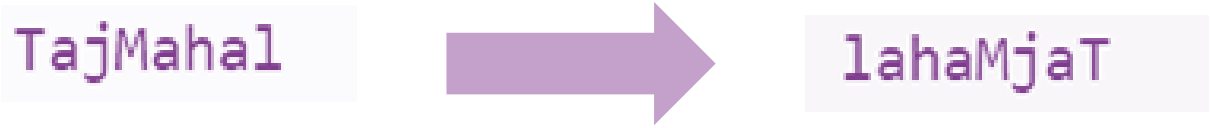
```
a appears 3 times in 'TajMahal'
```

۴۱- مرتب سازی string

simplyeasylearning

aaeegiillmnnprssyy

۴۲- معکوس کردن string



۴۳- مقایسه دو string

```
advise and advice are NOT identical
```

٤٤- کپی گرفتن از string

```
s1 = TajMahal
```

```
s2 = TajMahal
```

۴۵- ادغام دو string

```
char s1[10] = "Taj";
char s2[] = "Mahal";
```

```
TajMahal
```

٤٦- جستجوی یک کلمه در string

```
char s1[] = "Beauty is in the eye of the beholder";
char s2[] = "the";
```

```
'the' appears 2 time(s)
```

https://www.tutorialspoint.com/learn_c_by_examples/index.htm

https://www.tutorialspoint.com/data_structures_algorithms/index.htm

https://www.tutorialspoint.com/design_and_analysis_of_algorithms/index.htm

https://www.geeksforgeeks.org/sorting-algorithms/

https://www.w3schools.com/c/tryc.php?filename=demo_compiler

اصول و مبانی برنامه‌نویسی

مجید شبیری

کارشناسی ارشد IT، گرایش شبکه

از دانشگاه صنعتی امیرکبیر