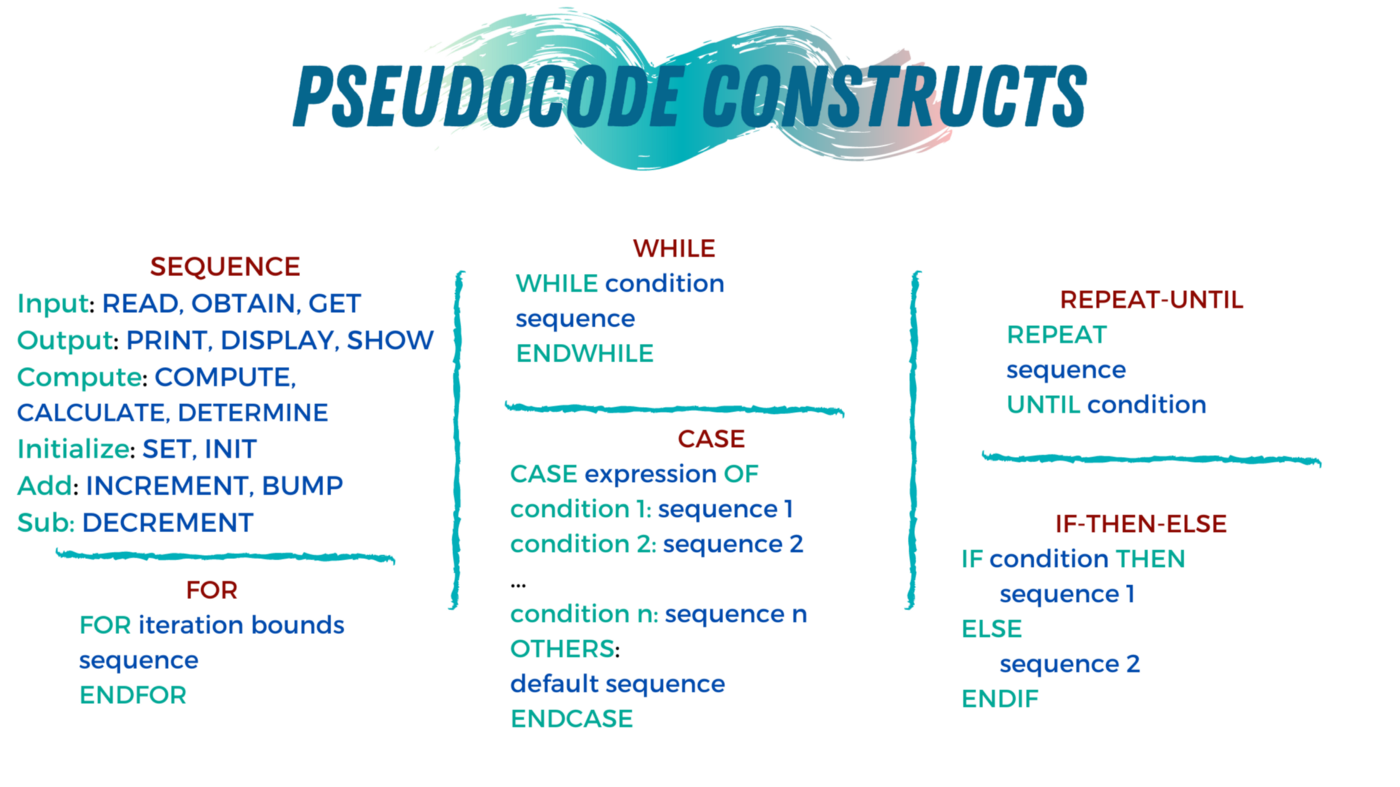
# The main constructs of pseudocode

The core of pseudocode is the ability to represent 6 programming constructs (always written in uppercase): SEQUENCE, CASE, WHILE, REPEAT-UNTIL, FOR, and IF-THEN-ELSE. These constructs — also called keywords —are used to describe the control flow of the algorithm.

1. **SEQUENCE** represents linear tasks sequentially performed one after the other.
2. **WHILE** a loop with a condition at its beginning.
3. **REPEAT-UNTIL** a loop with a condition at the bottom.
4. **FOR**another way of looping.
5. **IF-THEN-ELSE** a conditional statement changing the flow of the algorithm.
6. **CASE** the generalization form of IF-THEN-ELSE.



PseudoCode 2

**Path :**  
**Factorial**  
  
Write an algorithm to find the factorial of given number.

END

Factorial(number):

END WHILE

SET Fact = 1 and i = 1

SET i=i+1

WHILE i<=number

PRINT Fact

SET Fact=Fact\*i

Sol:

Factorial(number):

SET Fact = 1 and i = 1

WHILE i<=number

SET Fact=Fact\*i

SET i=i+1

END WHILE

PRINT Fact

END

PseudoCode 5

**Path :**

**Fibonacci Sequence**  
  
Write an algorithm to generate the Fibonacci Sequence upto to the given number.

SET first = 0 , second = 1 and i = 2

Fibonacci(number):

WHILE (i<number)

SET second = next and i = i+1

PRINT first and second

SET first = second

END

SET next = first + second and PRINT next

ENDWHILE

**Sol:**

Fibonacci(number):

SET first = 0 , second = 1 and i = 2

PRINT first and second

WHILE (i<number)

SET next = first + second and PRINT next

SET first = second

SET second = next and i = i+1

ENDWHILE

END

Pseudo\_Number of digits

**Path :**  
**Number of digits**  
  
Write an algorithm to display the number of digits in a given number.

NumberOfDigits(number):

ENDWHILE

PRINT count

SET count=0

WHILE (number > 0):

SET count=count+1 and SET number=number/10

END

**Sol:**

NumberOfDigits(number):

SET count=0

WHILE (number > 0):

SET count=count+1 and SET number=number/10

ENDWHILE

PRINT count

END

Pseudo\_Maximum element in an array

**Path :**  
**Maximum element in an array**  
  
Write an algorithm to display the maximum element in an array.

END

ENDWHILE

IF (arr[i]>max) THEN

ENDIF

WHILE (i<N):

SET i=1 and max=arr[0]

ArrayMaxElement(arr, N):

SET max=arr[i]

SET i=i+1

PRINT max

**Sol:**

ArrayMaxElement(arr, N):

SET i=1 and max=arr[0]

WHILE (i<N):

IF (arr[i]>max) THEN

SET max=arr[i]

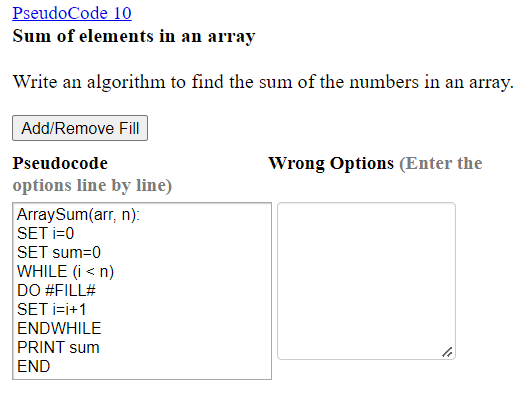
ENDIF

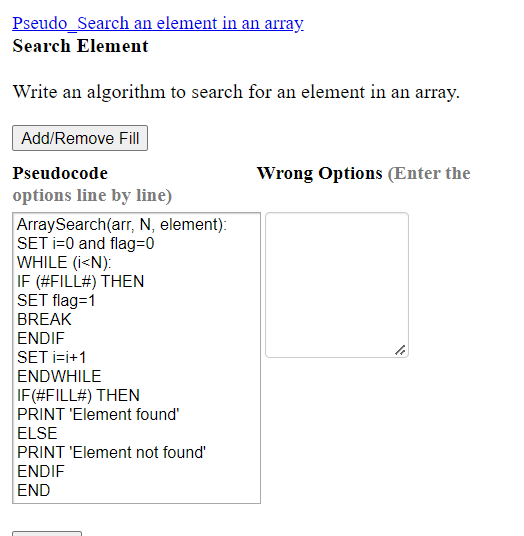
SET i=i+1

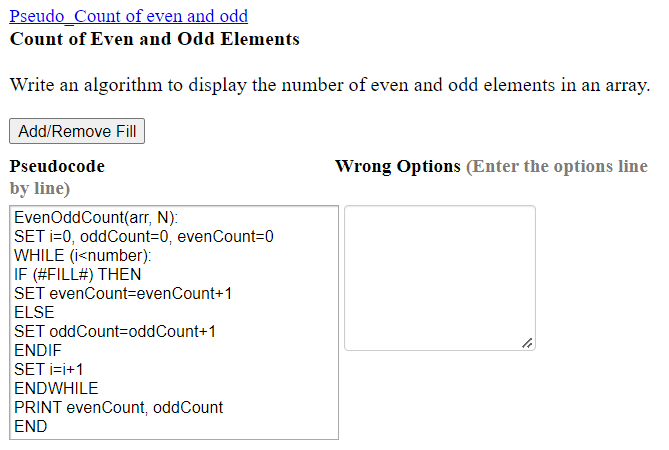
ENDWHILE

PRINT max

END







PseudoCode 12 - Maximum of Three numbers

**Path :**

Write an algorithm to find the maximum among three numbers.

Write n1 as greater

ELSE-IF: n2 > n3 THEN

Maximum Among three numbers:

IF: n1 > n2 AND n1 > n3 THEN

END-IF

Write n3 is greater

Read three numbers n1,n2,n3

Write n2 as greater

ELSE:

Pseudo\_Palindrome

**Path :**  
**Palindrome**  
  
Write an algorithm to check whether the given number is a palindromic number or not.

ENDIF

SET reverse = (reverse \* 10) + lastDigit and SET number = number / 10

END

SET lastDigit = number % 10

ENDWHILE

PRINT 'NOT A PALINDROME'

WHILE (number > 0):

ELSE

Palindrome(number):

SET reverse = 0 and temp=number

IF (temp == reverse) THEN

PRINT 'PALINDROME'

Pseudo\_Palindrome

**Path :**  
**Palindrome**  
  
Write an algorithm to check whether the given number is a palindromic number or not.

ENDIF

SET reverse = (reverse \* 10) + lastDigit and SET number = number / 10

END

SET lastDigit = number % 10

ENDWHILE

PRINT 'NOT A PALINDROME'

WHILE (number > 0):

ELSE

Palindrome(number):

SET reverse = 0 and temp=number

IF (temp == reverse) THEN

PRINT 'PALINDROME'

=========================Algolrithm========================

Algorithm for Prime Number:

Step 1: Start

Step 2: Initialize variables num,flag=1, j=2

Step 3: Read num from user

Step 4: If num<=1 // Any number less than 1 is not a prime number

Display "num is not a prime number"

Goto step 7

Step 5: Repeat the steps until j<[(n/2)+1]

5.1 If remainder of number divide j equals to 0,

Set flag=0

Goto step 6

5.2 j=j+1

Step 6: If flag==0,

Display num+" is not prime number"

Else

Display num+" n is prime number"

Step 7: Stop

Example to reverse an integer entered by the user in Pseudocode . This problem is solved by using loop in this example.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | OUTPUT "Please Enter any Number"  INPUT Number  Reverse = 0    WHILE(Number > 0)THEN      Reminder = Number %10      Reverse = (Reverse \*10) + Reminder      Number = Number // 10  END WHILE    OUTPUT "Reverse of entered number is ="+ Reverse |

# Factorial Program

## Algorithm

Algorithm of this program is very easy −

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

## Pseudocode

We can draft a pseudocode of the above algorithm as follows −

procedure find\_factorial(number)

FOR value = 1 to number

factorial = factorial \* value

END FOR

DISPLAY factorial

end procedure

## Implementation

Implementation of this algorithm is given below –

#include <stdio.h>

int main() {

int loop;

int factorial=1;

int number = 5;

for(loop = 1; loop<= number; loop++) {

factorial = factorial \* loop;

}

printf("Factorial of %d = %d \n", number, factorial);

return 0;

}

## Output

Output of the program should be −

Factorial of 5 = 120

## Program to count the total number of vowels and consonants in a string.

### Algorithm

1. Define a string.
2. Convert the string to lower case so that comparisons can be reduced. Else we need to compare with capital **(A, E, I, O, U)**.
3. If any character in string matches with vowels **(a, e, i, o, u )** then increment the vcount by 1.
4. If any character lies between 'a' and 'z' except vowels, then increment the count for ccount by 1.
5. Print both the counts.