Mathematical Operators:

- **Modulo operator** ( % ) gives the remainder of integer division of two values or variables. Make sure to divide the first value by the second value using integer division, so you can find the reminder.
  
  Example:
  
  \[ y = 7 \]
  
  \[ x = y \mod 2 \] here \( 7 / 2 = 3 \) with remainder of 1 therefore \( x = 1 \)

- **Division** ( / )
  
  The result of division depends on the type of variable
  
  Example:
  
  ```
  int y = 9 / 2
  y = 4 NOT y = 4.5 because y was defined as an integer ( int ). In memory the computer will store the integer part of the answer which 4.
  ```

- **Assignment operator:**
  
  - **Assignment operator** ( = ) assigns a value to a variable:
  
  Example:
  
  ```
  x = 2 NOT 2 = x
  x = y / 2 NOT y /2 = x
  x = y means x will be assigned the value of y BUT y = x means y will be assigned the value of x
  ```

  The variable is always on the left of the assignment operator ( = ), and the value you are trying to assign to that variable is on right.

  - **Compound assignment** ( +=, -=, *=, /=, %= ): When using compound assignment operators, make sure to write the mathematical operator first then the assignment.

  Example:
  
  ```
  x = x + y can be written as x += y NOT x =+ y
  ```
Pseudocode / Programming:

- **Print:**
  When printing the value of a variable (the content of the variable in memory), **DO NOT** use " "
  Example:
  ```
  print sum  **NOT** print "sum"
  ```

  If you want to print text, use " "
  Example:
  ```
  print "Hello, this is my first program"
  ```

- **cout<<**
  When writing c++ code use cout<< **NOT** print.
  Example:
  ```
  cout<<sum ; **NOT** print sum ;
  ```

  If you want to print text use " "
  Example:
  ```
  cout<<"Hello, this is my first program" ;
  ```
- **While loop:**

```
while ( loop test )
{
    statements
}
```

**Things you need to check when writing a while loop:**

1) **Make sure to initialize all the variables before the loop.**

Example:
```
sum = 0
x = 1
product = 1
while ( x > 100)
{
}
```

2) **Check the loop test to avoid infinite loop (loop that runs for ever and never stops)**

Example 1:
```
sum = 0
x = 1
while ( x >= 1)
{
    sum = sum + x
    x ++
}
```

Since x starts with 1 and keep incrementing inside the loop, this condition will always be true, and this loop will never stop.

Example 2:
```
sum = 0
x = 1
while ( x >= 1)
{
    sum = sum + x
}
```
Another common mistake with loops occurs when you don’t change or update the value of the test variable inside the loop. The condition to stop executing the loop will be always true and this loop will never stop.

3) **Check the loop test to make sure that the loop will be executed.**

Example:

```plaintext
sum = 0
x = 0
while ( x >= 1)
{
    sum = sum + x
    x++;
}
```

Since x starts with zero this loop will not be executed x is not >= 1

4) **Check where to print the variables inside or outside the loop.**

Example 1: Print the sum of all integers from 1 to 100 → outside the loop

```plaintext
sum = 0
x = 1
while ( x <= 100)
{
    sum = sum + x
    x++;  
}
print sum
```
Example 2: Print all the even integers from 1 to 100 → inside the loop

\[
x = 1
\]

while ( x <= 100)
{
    if ( x % 2 == 0)
    {
        print x
    }
    x++;
}
Pseudocode / Programming:

- **for loop:**

  ```
  for ( variables initialization ; loop test ; update the value of the test variable )
  {
    statements
  }
  ```

**Things you need to check when writing for loop:**

**5) Make sure to follow the proper format of a for loop.**

Example:

```
sum = 0
x = 1
for ( x <=100)  ← this is NOT correct format of for loop
{
  sum = sum + x
  x ++
}
print sum
```

The correct way to write the for loop:

```
sum = 0
for ( x = 1 ; x <= 100 ; x ++) ← for (initialize ; test ; update)
{
  sum = sum + x
}
print sum
```

**6) Make sure NOT to update the test variable twice**

Example 1:

```
sum = 0
for ( x = 1 ; x <= 100 ; x ++)
{
  sum = sum + x
  x++ ← this is NOT correct as you update the value of x twice
}
print sum
```
7) Check the loop test to make sure that the loop will be executed, and that it will not be an infinite loop.

Example:

```
sum = 0
for ( x = 1 ; x >= 1 ; x ++)
{
    sum = sum + x
}
print sum
```

as x starts with 1 and keep incrementing (x++) each iteration, it will always have a value > 1. Therefore, this for loop will never stop.

8) Check where to print the variables inside or outside the loop.

Example 1: Print the sum of all integers from 1 to 100 → outside the loop

```
sum = 0
for ( x = 1 ; x <= 100 ; x ++)
{
    sum = sum + x
}
print sum
```

Example 1: Print all the even integers from 1 to 100 → inside the loop

```
for ( x = 1 ; x <= 100 ; x ++)
{
    if ( x % 2 == 0)
    {
        print x
    }
}
```
Common notes about writing Pseudocode / Programming:

- Read the question carefully and follow the instructions. While reading try to identify:
  - What are the variables you need to define, and what are the starting values for these variables?
  - What does the question ask you to solve?
  - What you should print?

Example:
Write pseudocode to print the sum for all the integers from 1 to 45000.

- **What are the needed variables?**
  
  sum = 0
  x = 1

- **What to solve?** Calculate the sum for all the integers from 1 to 45000

  while ( x <= 45000)
  {
    sum = sum + x
    x ++
  }

- **What you should print?** Print sum

  print sum

Note: Calculate the sum only when the program asks you to calculate the sum.