

```

print("Hello World!")

Hello World!

a = 10
b = 5

# Addition
print("Addition:", a + b)

# Subtraction
print("Subtraction:", a - b)

# Multiplication
print("Multiplication:", a * b)

# Division
print("Division:", a / b)

#Floor Division
print("Floor Division:", a//b)

#Modulo
print("Modulo:", a%b)

#Exponential
print("Exponential:", a**b)

```

```

Addition: 15
Subtraction: 5
Multiplication: 50
Division: 2.0
Floor Division: 2
Modulo: 0
Exponential: 100000

```

```

win_points = 3
draw_points = 1
lost_points = 0

total_points = (6 * win_points) + (1 * draw_points) + (2 * lost_points)

print("Total points scored:", total_points)

```

```

Total points scored: 19

```

```

r = 50
a = 60
ap = 20
total = r + a + ap
print("r percentages: ", r/ total * 100)
print("a percentages: ", a/ total * 100)
print("ap percentages: ", ap/ total * 100)
print(total)
if(r == a):
    print("true")
else:
    print("false")

```

```

r percentages: 38.46153846153847
a percentages: 46.15384615384615
ap percentages: 15.384615384615385
130
false

```

You're creating a program to manage a zoo's animal population. Declare a variable `lion_population` with an initial value of 10. The zoo welcomes 5 new lion cubs. Update the `lion_population` variable and print the total lion population.

```
lion_population = 10
lion_population += 5
print("Total lion population:", lion_population)
```

➞ Total lion population: 15

You're developing a weather monitoring system. Declare a variable temperature with a value of 25.5 degrees Celsius. Due to a sudden heatwave, the temperature increases by 8 degrees. Update and print the new temperature.

```
temperature = 25.5
temperature += 8
print("Updated temperature:", temperature)
```

➞ Updated temperature: 33.5

A science experiment involves tracking the growth of a plant. Declare a variable plant_height with an initial value of 15 centimeters. Over a week, the plant grows 2.5 centimeters taller each day. Update and print the final height of the plant after the week.

```
plant_height = 15
growth_per_day = 2.5
num_days = 7

final_height = plant_height + (growth_per_day * num_days)

print("Final height of the plant after a week:", final_height, "centimeters")
```

➞ Final height of the plant after a week: 32.5 centimeters

You're designing a space mission trajectory. Declare variables initial_velocity and acceleration with values 3000 meters per second and 500 meters per second squared respectively. Calculate and print the final velocity after 10 seconds

```
initial_velocity = 3000
acceleration = 500
time = 10

final_velocity = initial_velocity + (acceleration * time)

print("Final velocity after 10 seconds:", final_velocity, "meters per second")
```

➞ Final velocity after 10 seconds: 8000 meters per second

A group of friends is sharing a pizza. Declare a variable pizza_slices with a value of 8. Each friend wants to have an equal number of slices, and there are 5 friends. Calculate and print the maximum number of slices each friend can have without cutting the pizza.

```
pizza_slices = 8
num_friends = 5

max_slices_per_friend = pizza_slices // num_friends

print("Maximum number of slices each friend can have:", max_slices_per_friend)
```


➞ Maximum number of slices each friend can have: 1

You're modeling the movement of a pendulum. Declare a variable pendulum_length with a value of 1.2 meters. Calculate and print the period of oscillation (time taken for one complete swing) using the formula ($T = 2\pi \sqrt{\frac{L}{g}}$), where (π) is pi (approximately 3.14159) and (g) is the acceleration due to gravity (approximately (9.81) meters per second squared).

```
import math

pendulum_length = 1.2
gravity = 9.81
pie = 3.14159
period = 2 * math.pi * math.sqrt(pendulum_length / gravity)
# period = 2 * pie * (pendulum_length/gravity)**0.5

print("Period of oscillation:", period, "seconds")
```


 Period of oscillation: 2.1975359457694705 seconds

A software company is tracking the number of bugs in their codebase. Declare a variable `bug_count` with an initial value of 100. After a round of debugging, 35 bugs are fixed. Update and print the new `bug_count`.

```
bug_count = 100
fixed_bugs = 35

bug_count -= fixed_bugs

print("New bug count:", bug_count)
```

 New bug count: 65

You're building a game where players collect gems. Declare a variable `gem_count` with an initial value of 50. Each time a player finds a gem, 5 gems are added to their collection. Update and print the new `gem_count`

```
gem_count = 50
gems_collected = 5

gem_count += gems_collected

print("New gem count:", gem_count)
```

 New gem count: 55

A fitness tracker is monitoring a user's heart rate variability (HRV). Declare a variable `hrv_index` with an initial value of 80. After a relaxation session, the user's HRV improves by 10 points. Update and print the new `hrv_index`.

```
hrv_index = 80
hrv_improvement = 10

hrv_index += hrv_improvement

print("New HRV index:", hrv_index)
```

 New HRV index: 90


You're simulating the growth of a bacterial colony. Declare a variable `bacteria_count` with an initial value of 5000. Over a day, the colony doubles in size every 4 hours. Update and print the new `bacteria_count`.

```
bacteria_count = 5000
doubling_time = 4

num_doublings = 24 / doubling_time

new_bacteria_count = bacteria_count * 2**num_doublings

print("New bacteria count:", new_bacteria_count)
```

 New bacteria count: 320000.0

Start coding or [generate](#) with AI.

