# EXAMPLES OF CODE SYNTAX



## PRINT & INPUT

| | |
|---|---|
| PRINT | print(“This prints to the console”) |
| INPUT | answer = input(“What is the value?”) |

## DATA TYPES

| | |
|---|---|
| STRING | “This is a string” |
| INTEGER | 4 |
| FLOAT | 3.14 |
| BOOLEAN | True |
| TYPE | type(“4”) |
| CAST TO STRING | str(4) |
| CAST TO INTEGER | int(“4”) |
| CAST TO FLOAT | float(“3.14”) |
| CAST TO BOOLEAN | bool(0) |

## STRINGS

| | |
|---|---|
| CONCATENATE STRINGS | “a” + “b” + “c” |
| MULTIPLY STRINGS | “a” * 3 |
| LENGTH | len(“This is a string”) |
| CHARACTER | my_string[2] |
| LOWER CASE | my_string.upper() |
| UPPER CASE | my_string.lower() |
| SPLIT | my_string.split(“,”) |
| REPLACE | my_string.replace(“c”,”d”) |
| MAX | max(my_string) |
| MIN | min(my_string) |
| FIND | my_string.find(“b”) |
| REVERSE FIND | my_string.rfind(“b”) |
| SPLICE | my_string[1:3] |
| IN | “a” in my_string |

## ASSIGNMENT OPERATORS

| | |
|---|---|
| ASSIGN VALUE | my_var = 10 |
| INCREMENT | my_var += 1 |
| DECREMENT | my_var -= 1 |
| MULTIPLY ASSIGNMENT | my_var *= 4 |
| DIVIDE ASSIGNMENT | my_var /= 2 |

**MATH OPERATORS**

| | |
|---|---|
| ADDITION | 6 + 3 |
| SUBTRACTION | 6 - 3 |
| MULTIPLICATION | 6 * 3 |
| DIVISION | 6 / 3 |
| FLOOR DIVISION | 6 // 3 |
| MODULO | 6 % 3 |
| EXPONENTS | 6 ** 3 |
| ABSOLUTE VALUE | abs(-24) |
| ROUND | round(3.1415, 2) |

**COMPARISONS & LOGIC**

| | |
|---|---|
| EQUAL TO | a == b |
| GREATER THAN | a > b |
| GREATER OR EQUAL | a >= b |
| LESS THAN | a < b |
| LESS OR EQUAL | a <= b |
| NOT EQUAL TO | a != b |
| TRUE | True |
| FALSE | False |
| AND | True and False |
| OR | True or False |
| NOT | not True |

**LISTS**

| | |
|---|---|
| CREATE LIST | my_list = [“a”, “b”, “c”, “d”] |
| GET VALUE | my_list[2] |
| GET INDEX | my_list.index(“b”) |
| SORT | my_list.sort() |
| REVERSE | my_list.reverse() |
| LENGTH | len(my_list) |
| APPEND | my_list.append(“e”) |
| REMOVE | my_list.remove(“d”) |
| INSERT | my_list.insert(“e”, 2) |
| COUNT | my_list.count(“a”) |
| MAX | max(my_list) |
| MIN | min(my_list) |
| FIND | my_list.find(“b”) |
| REVERSE FIND | my_list.rfind(“b”) |
| SPLICE | my_list[1:3] |
| IS | my_list is [“a”, “b”, “c”, “d”] |
| IN | “value” in my_list |

**LOOPS**

FOR LOOP

```
for i in range(5):
    print(i)
```

FOR LOOP WITH LIST

```
my_list = [0, 1, 2, 3, 4]
for item in my_list:
   print(item)
```

FOR LOOP WITH STRING

```
my_string = “hello”
for letter in my_string:
    print(letter)
```

WHILE LOOP

```
x = 0
while x < 5:
    print(x)
    x +=1
```

BREAK

```
break
```

CONTINUE

```
continue
```

**IF ELIF ELSE**

IF STATEMENT

```
if answer == “yes”:
    print(“Correct”)
```

IF ELSE

```
if answer == “yes”:
    print(“Correct”)
else:
    print(“Incorrect”)
```

IF ELIF ELSE

```
if temp > 100:
    print(“Too high”)
elif temp < 90:
    print(Too low”)
else:
    print(“Just right”)
```

**RANDOM MODULE**

IMPORT

```
import random
```

RANDOM FLOAT

```
random.random()
```

RANDOM INTEGER

```
random.randint(1,10)
```

RANDOM CHOICE FROM LIST

```
random.choice(my_list)
```

**MATH MODULE**

| | |
|---|---|
| IMPORT | import math |
| SQUARE ROOT | math.sqrt(9) |
| PI | math.pi |
| E | math.e |
| NATURAL LOG | math.log(100) |
| LOG BASE 10 | math.log(100, 10) |
| RADIANS | math.radians(90) |
| SINE | math.sin(angle) |
| COSINE | math.cos(angle) |
| TANGENT | math.tan(angle) |

**FUNCTIONS**

| | |
|---|---|
| DEFINE FUNCTION | def my_function():<br>  print(“my_function is running”) |
| DEFINE WITH PARAMETERS | def my_function(param1, param2):<br>  print(param1)<br>  print(param2) |
| RETURN STATEMENT | return my_result |
| CALL FUNCTION | my_function() |
| CALL WITH ARGUMENTS | my_function(“value1”, “value2”) |
| CALL AND STORE RETURN | result = my_function() |