1 Variables

In Python, variables are declared and initialized by simply assigning a value to it. The type of the variable is determined by the value assigned to it. For example, the following code declares and initializes a variable named x to the value 5. The type of the variable x is an integer.

```
x = 5 # x is an integer
```

1.1 Rules of naming variables

Variables must be named according to the following rules:

- Variable names must start with a letter or an underscore.
- The remainder of your variable name may consist of letters, numbers and underscores.
- Names are case sensitive.
- Names must not be a reserved word.

Here are some examples of legal and illegal variable names:

```
# Examples of valid variable name
name = "Bob"
_my_name = "Bob"
myName = "Bob"
MYNAME = "Bob" # Note: myName and MYNAME are different variables
age = 23
height = 1.73
is_human = True
name2 = "Alice"

# Examples of invalid variable name
2name = "Bob"
my-name = "Bob"
my name = "Bob"
```

1.2 Reserved Words

The following words are reserved by Python and cannot be used as variable names:

False	class	from	or
None	continue	global	pass
True	def	if	raise
and	del	import	return
as	elif	in	try
assert	else	is	while
async	except	lambda	with
await	finally	nonlocal	yield
break	for	not	

2 Python Data Types

Here are some basic data types in Python:

- Text Type: str
- Numeric Types: int, float
- Boolean Type: bool

2.1 Text Type

Type	Description	Example
str	String literals in python are surrounded by either single quotation marks, or double quotation marks.	'hello' or "hello"

2.2 Numeric Types

Type	Description	Example
int	Integers is a whole number, positive or negative, without decimals, of unlimited length.	x = 1
float	Float is a number, positive or negative, containing one or more decimals.	x = 1.0

2.3 Boolean Type

Type	Description	Example
bool	Booleans represent one of two values: True or False.	x = True

2.4 Type Conversion

You can convert from one type to another with the int(), float(), and str() methods.

Method	Description	Example	Result
int()	Converts a string to an integer.	x = int("1")	x = 1
float()	Converts a string to a float.	x = float("1.0")	x = 1.0
str()	Converts an integer to a string.	x = str(1)	x = "1"

3 Operators

In this section, x and y are assigned as follow:

```
x = 15
y = 2
```

3.1 Arithmetic Operator

Operator	Description	Example	Result
+	Addition	x + y	17
-	Subtraction	x - y	13
*	Multiplication	x * y	30
/	Division	x / y	7.5
%	Modulus	x % y	1
//	Floor Division	x // y	7
**	Exponentiation	x ** y	225

3.2 Assignment Operator

Operator	Description	Example	Result
=	Assign	x = 5	x = 5
+=	Add	x += 5	x = 20
-=	Subtract	x -= 5	x = 10
*=	Multiply	x *= 5	x = 75
/=	Divide	x /= 5	x = 3
%=	Modulus	x %= 5	x = 0
//=	Floor Division	x //= 5	x = 3
**=	Exponentiation	x **= 5	x = 759375

3.3 Comparison Operators

Comparison Operators (a.k.a Relational Operators) are used to compare values and create a conditional statement. It either returns True or False according to the condition.

Operator	Description	Example	Result
==	Equal	x == y	False
!=	Not Equal	x != y	True
>	Greater Than	x > y	True
<	Less Than	x < y	False
>=	Greater Than or Equal to	x >= y	True
<=	Less Than or Equal to	x <= y	False

3.4 Logical Operators

Logical operators are used to combine conditional statements. It either returns True or False according to the condition.

Operator	Description	Example	Result
and	Returns True if both statements are true	x < 5 and x < 10	True
or	Returns True if one of the statements is true	x < 5 or x < 4	True
not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)	False

4 Conditions

4.1 If Statement

If Statement is used to check if a condition is true or false. If the condition is true, it will execute the code inside the if statement. If the condition is false, it will not execute the code inside the if statement.

Syntax

```
if condition:
    statement # The statement will execute if the condition is true
```

Example

```
A = 100
if A > 0:
    print("A is positive") # Note, This line is indented
```

4.2 If...Else Statement

If...Else Statement is used to check if a condition is true or false. If the condition is true, it will execute the code inside the if statement. If the condition is false, it will execute the code inside the else statement.

Syntax

```
if condition:
    statement # The statement will execute if the condition is true
else:
    statement # The statement will execute if the condition is false
```

Example

```
A = 100
if A > 0:
    print("A is positive") # Note: This line is indented
else:
    print("A is negative") # Note: This line is indented
```

4.3 If...Elif...Else Statement

If...Elif...Else Statement is used to check if a condition is true or false. If the condition is true, it will execute the code inside the if statement. If the condition is false, it will check the next condition. If the next condition is true, it will execute the code inside the elif statement. If the next condition is false, it will execute the code inside the else statement.

Syntax

```
if condition:
    statement # The statement will execute if the condition is true
elif condition:
    statement # The statement will execute if the condition is true but the
   first condition is false
else:
    statement # The statement will execute if the condition is false
```

Example

```
A = 100
if A > 0:
    print("A is positive") # Note: This line is indented
elif A == 0:
    print("A is zero") # Note: This line is indented
else:
    print("A is negative") # Note: This line is indented
```

4.4 Nested If Statement

A nested if statement contains an if statement inside another if statement. The inner if statement will be executed if the outer if statement is true.

Syntax

```
if condition:
    if condition:
        statement # The statement will execute if all the conditions are true
```

```
A = 100
if A > 0:
    print("A is positive")
    if A > 50:
        print("A is greater than 50")
```

4.5 Reminder

- The if statements are top down approach (if the first condition is true, it will not check the next condition)
- The condition of if/elif statements must not be empty
- All the statement must be indented
- The size of indentation does not matter, but all the indentations must be the same size
- The if statement can be used without else statement (but not vice versa)
- The elif statement can be used without else statement (but not vice versa)

5 User Input

5.1 Input Function

The input function is used to get input from the user. The input function will return a string value.

Syntax

```
variable = input(Message) # The message will be displayed to the user, but it is optional
```

Example

```
# This program is a simple example of how to use the input function
# to get the user's name and say hi with his/her name
name = input("What is your name? ")
print("Hello", name)
```

5.2 Multiple Input

There are different methods to get multiple inputs from the user. Here one of the methods.

Syntax

```
variable1, variable2 = input(Message).split() # The message will be displayed to the user, but it is optional
```

Example

```
# This program is an example of multiple input
name, age = input("What is your name and age? ").split()
print("Hello", name, "you are", age, "years old")
```