Learn Programming Easily

PYTHON

TEACHER-HTUN AUNG KYAW

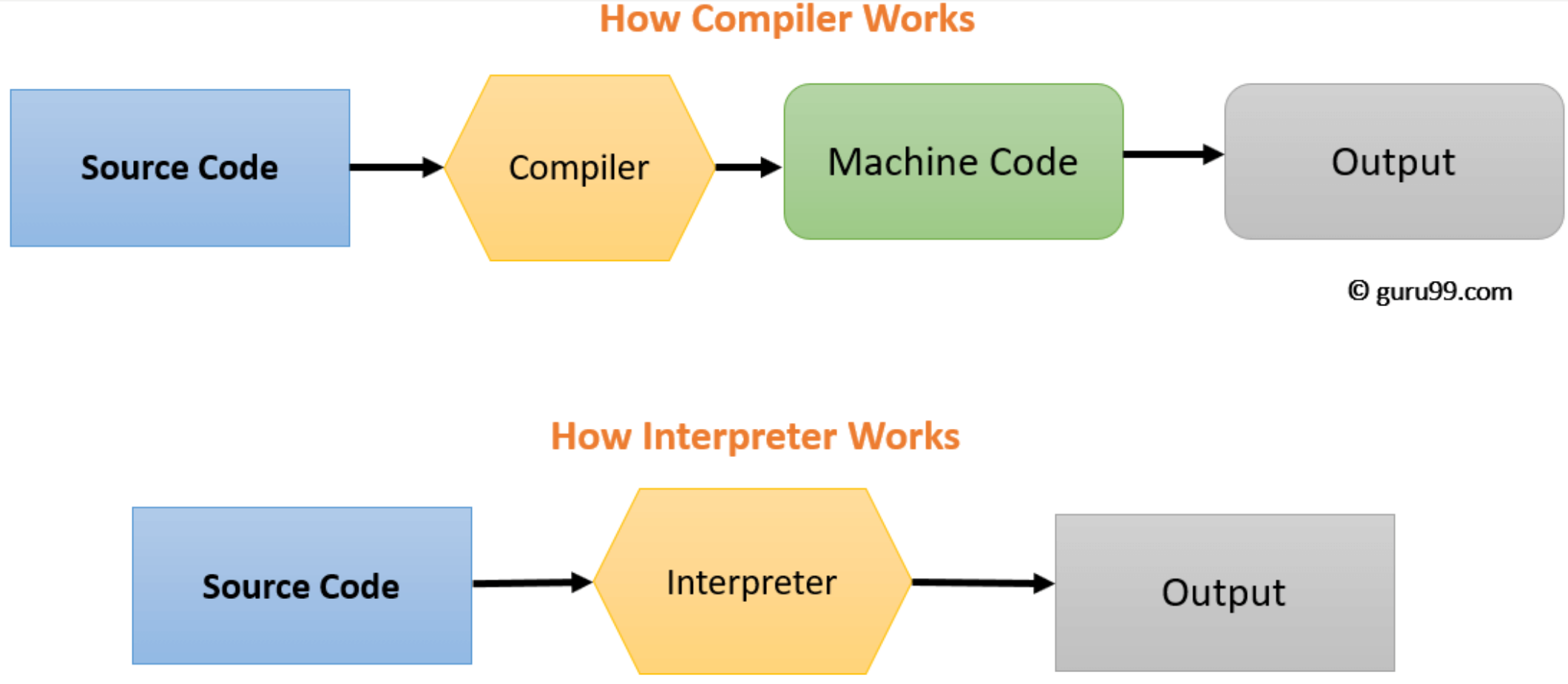
ACKNOWLEDGEMENT

- I wish to express my sincere thanks and appreciation to all of my teachers and the books and online classes which give me virtual learning space for programming.
- Especially Books by Deitel and Charles Severance Ph.D from University of Michigan

WHY PYTHON?

- Python is easy to use.
- Less syntax than any other programming languages
- Widely used today in various fields like science, AI, Web
- To find out more, visit python.org at
- https://www.python.org/about/apps/

INTERPRETER



PRIMITIVE DATA TYPE AND RESERVED WORDS

- The three main data type: int, float and str.
  There are a few more like tuple, list, dictionary.

- These are reserved words.

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

OPERATION

Python operation	Arithmetic operator	Algebraic expression	Python expression
Addition	`+`	$f + 7$	`f + 7`
Subtraction	`-`	$p - c$	`p - c`
Multiplication	`*`	bm	`b * m`
Exponentiation	`**`	x^y	`x ** y`
Division	`/` `//` (new in Python 2.2)	x / y or $\frac{x}{y}$ or $x \div y$	`x / y` `x // y`
Modulus	`%`	$r \bmod s$	`r % s`

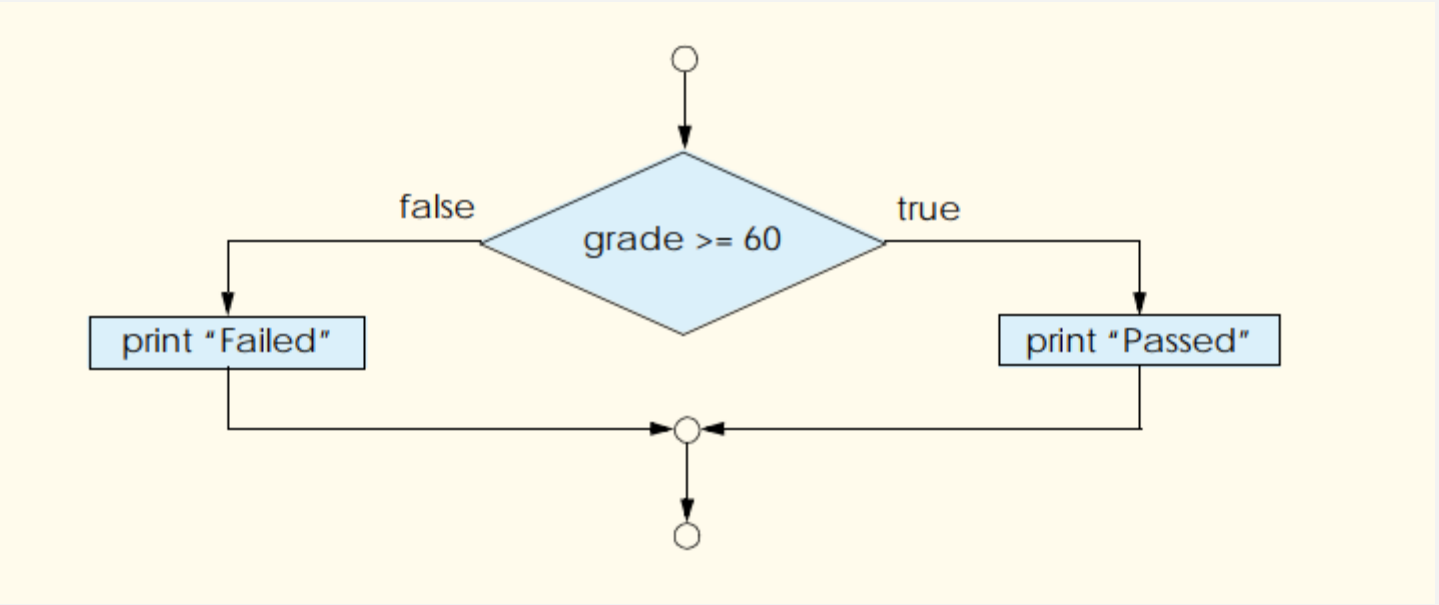
RELATIONAL AND EQUALITY OPERATOR

Standard algebraic equality operator or relational operator	Python equality or relational operator	Example of Python condition	Meaning of Python condition
Relational operators			
>	`>`	`x > y`	`x` is greater than `y`
<	`<`	`x < y`	`x` is less than `y`
≥	`>=`	`x >= y`	`x` is greater than or equal to `y`
≤	`<=`	`x <= y`	`x` is less than or equal to `y`
Equality operators			
=	`==`	`x == y`	`x` is equal to `y`
≠	`!=`, `<>`	`x != y`, `x <> y`	`x` is not equal to `y`

PRECEDENCE

Operators	Associativity	Type
`()`	left to right	parentheses
`**`	right to left	exponentiation
`*  /  %`	left to right	multiplicative
`+`	left to right	additive
`<  <=  >  >=`	left to right	relational
`==  !=  <>`	left to right	equality
`and`	left to right	logical AND
`or`	left to right	logical OR
`not`	right to left	logical NOT

CONDITION

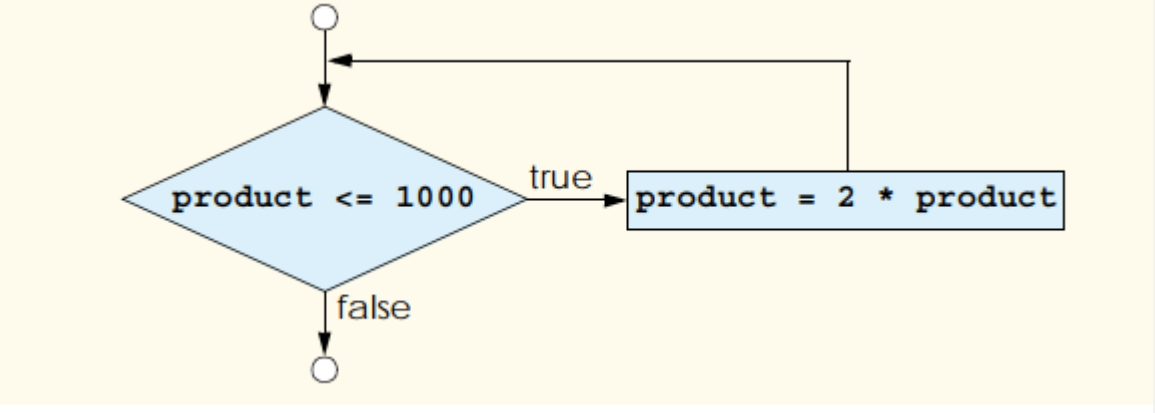


LOOPS

For loop

```
for i in range(1, 11, 1):
    print(i)
```

While loop



TYPE CONVERSION


```
num = input('Enter a number: ')
try:
    i_num = int(num)
except:
    i_num = 'a'

if i_num != 'a':
    print('You enter: ', num)
else:
    print('Not a number')
```

```
n = input("Enter name: ")

name = str(n)

print(name)

print(type(name))
```

FUNCTION

Method	Description	Example
`acos( x )`	Trigonometric arc cosine of x (result in radians)	`acos( 1.0 )` is `0.0`
`asin( x )`	Trigonometric arc sine of x (result in radians)	`asin( 0.0 )` is `0.0`
`atan( x )`	Trigonometric arc tangent of x (result in radians)	`atan( 0.0 )` is `0.0`
`ceil( x )`	Rounds x to the smallest integer not less than x	`ceil( 9.2 )` is `10.0` `ceil( -9.8 )` is `-9.0`
`cos( x )`	Trigonometric cosine of x (x in radians)	`cos( 0.0 )` is `1.0`
`exp( x )`	Exponential function e^x	`exp( 1.0 )` is `2.71828` `exp( 2.0 )` is `7.38906`
`fabs( x )`	Absolute value of x	`fabs( 5.1 )` is `5.1` `fabs( -5.1 )` is `5.1`
`floor( x )`	Rounds x to the largest integer not greater than x	`floor( 9.2 )` is `9.0` `floor( -9.8 )` is `-10.0`
`fmod( x, y )`	Remainder of x/y as a floating point number	`fmod( 9.8, 4.0 )` is `1.8`

`hypot( x, y )`	hypotenuse of a triangle with sides of length x and y: sqrt($x^2 + y^2$)	`hypot( 3.0, 4.0 )` is `5.0`
`log( x )`	Natural logarithm of *x* (base *e*)	`log( 2.718282 )` is `1.0` `log( 7.389056 )` is `2.0`
`log10( x )`	Logarithm of *x* (base 10)	`log10( 10.0 )` is `1.0` `log10( 100.0 )` is `2.0`
`pow( x, y )`	*x* raised to power *y* (x^y)	`pow( 2.0, 7.0 )` is `128.0` `pow( 9.0, .5 )` is `3.0`
`sin( x )`	trigonometric sine of *x* (*x* in radians)	`sin( 0.0 )` is `0.0`
`sqrt( x )`	square root of *x*	`sqrt( 900.0 )` is `30.0` `sqrt( 9.0 )` is `3.0`
`tan( x )`	trigonometric tangent of *x* (*x* in radians)	`tan( 0.0 )` is `0.0`

LEARN PROGRAMMING EASILY

```
import math as m

print(2*m.log(10, 10))

print(m.sqrt(400))
```

```
def add(a, b):
    return a+b


a = input("Enter a number: ")
num1 = int(a)
b = input("Enter a number: ")
num2 = int(b)

print("The result is: ", add(num1, num2))
```

Programmer-defined Function

DEFAULT ARGUMENT AND * ARGUMENT

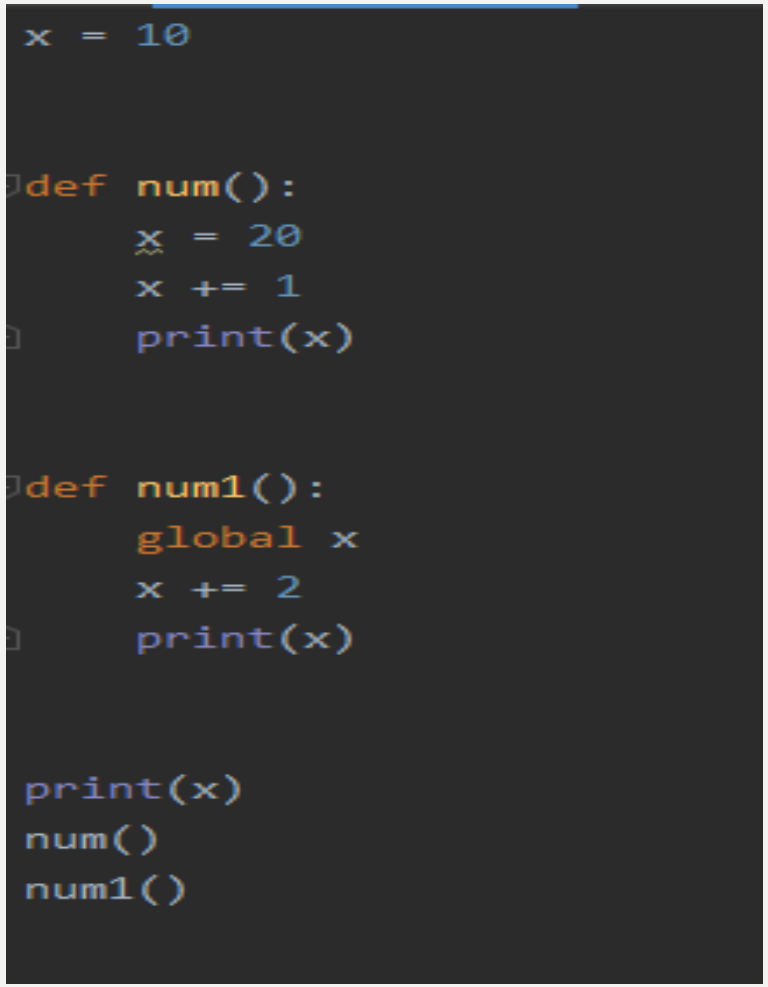

```
def area(length=1, width=2):
    return length*width


print("Area1", area())
print("Area2", area(10))
print("Area3", area(10, 3))
```

```
def names(*name):
    for n in name:
        print(n)


names("Tony", "Rogers", "Banner")
```

GLOBAL VS LOCAL



global keyword is used to access the global variable.

STRING

- String has so many inbuilt methods
- Check it out at python.org

```
name = "Htun Aung"

print(name[0])
print(name[0:4])

print(name.lower())
```

Upper(), Find(), Replace()

REFERENCES

- Python How to Program by Deitel
- University of Michigan: Charles Severance Ph.D : Python for Everybody