CMPE 102
Programming Logic and Design

Module 7
Python's Pre-defined Function

www.fppt.info

Outline

- Subroutine concept
- Built-in functions and standard modules
- Composition
- Defining functions
- Parameters and arguments
- Seeking and providing help

What are subroutines?

- A subroutine is a sequence of one or more actions grouped into a single task
  - The task won't be performed until the subroutine itself is used

This button won't do anything until it is pushed

- Subroutines are also known by other names, like

subroutines

methods

functions

procedures

www.fppt.info

Have you seen Python functions?

- Yes, you have

```
print(82+64+90+75+33)
```

```
print('Hello')
```

```
val = input()
```

argument

Some functions may contain no argument

- These are parts of the Python’s *built-in functions*—they are readily available to use
- Functions are always followed by parentheses
- Within parentheses, it contains *arguments* of the function
  - A function may have no argument or 1 or more argument

www.fppt.info

Essential built-in functions

- Some examples of built-in functions:

```
abs()      float()    input()    int()
len()      list()     max()      pow()
print()    range()    round()    sum()
type()
```

- No need to remember all of these now
  - We will eventually learn how to use them later
- For each function, you need to learn how to use it, i.e. what argument(s) to send

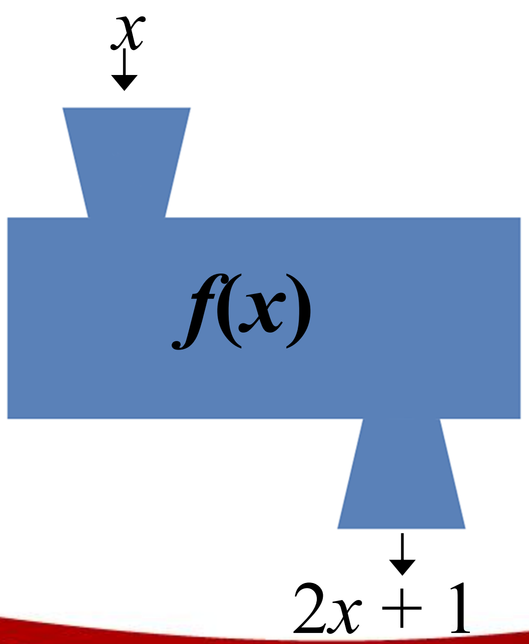
Function – How does it work?

- Function usually has *input(s)* and/or *output(s)*
- For example, in math, suppose you have a function `f(x)` defined as follow

Note: This one is math, not Python!

$$f(x) = 2x + 1$$

- This means
  - – x is an input to the function $f(x)$
  - – The function produces a result

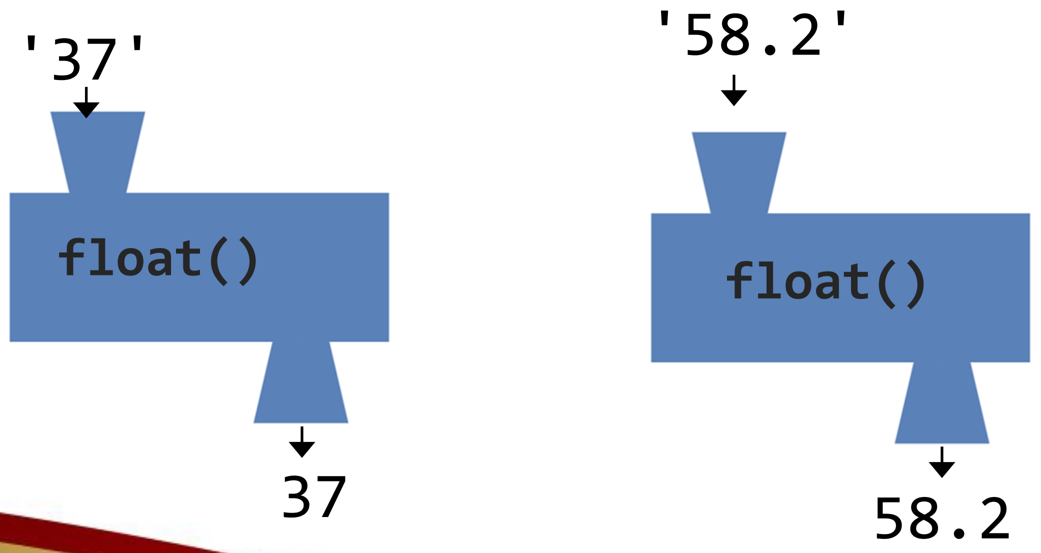


www.fppt.info

Calling a Function

- In Temperature Conversion task, we *call* the function `float()` to convert a string to a number

```
celsius_str = input()
celsius = float(celsius_str)
```



www.fppt.info

Task: Phone Bill

- Long-distance rate for a domestic call is Php 2/minute, while a fraction of a minute is charged as a whole minute
- For example
  - 1-minute call → Php2
  - 3-minute call → Php5
  - 5.2-minute call → Php12
- Write a program that
  - asks the user how many seconds is used for the call
  - then computes the total charge for the call

www.fppt.info

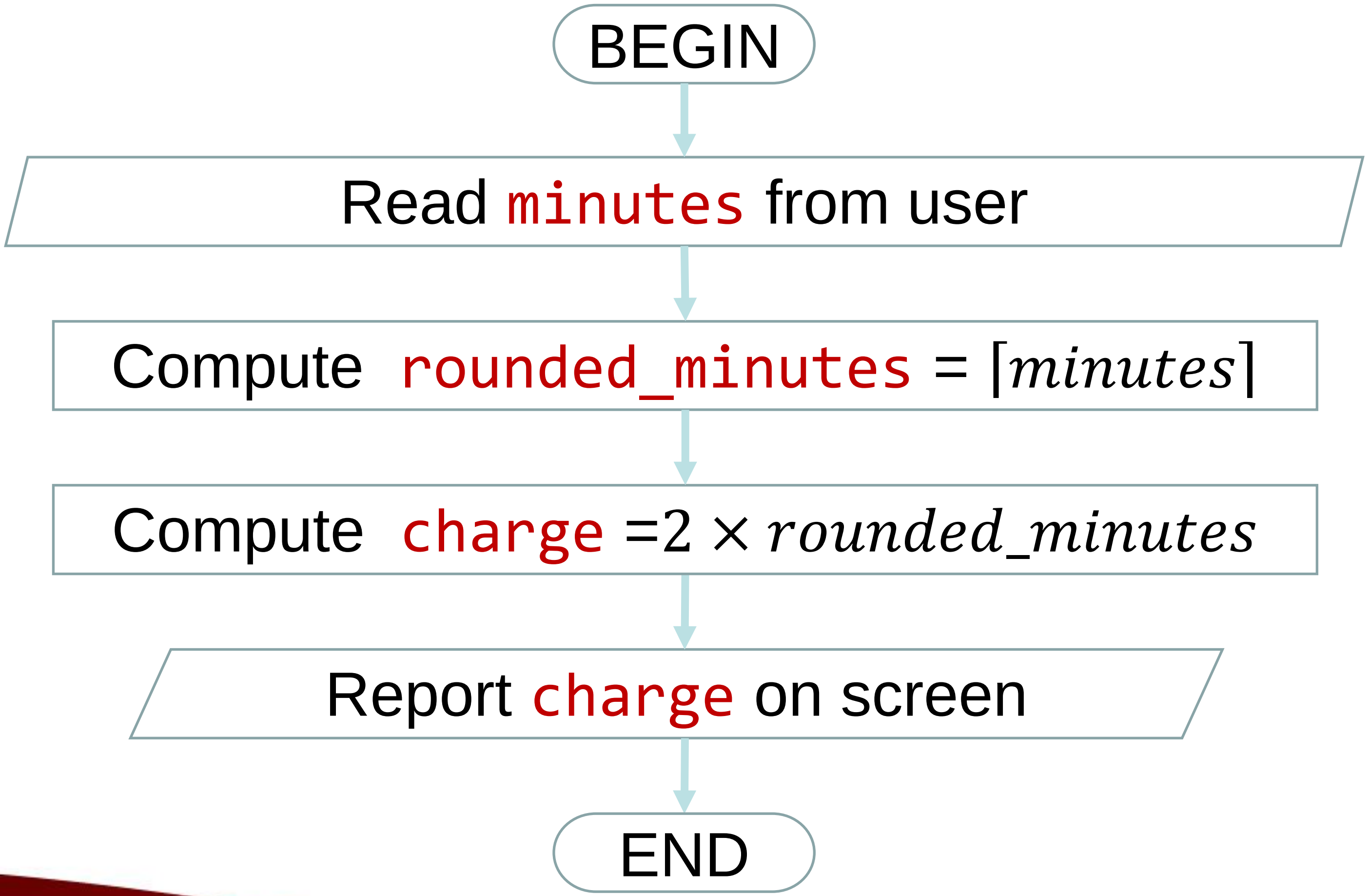
Phone Bill - Ideas

- At first, the problem looks like a typical division problem
- However, the fraction of the result must not be discarded this time, but will be rounded up to the nearest integer
  - E.g., 3 is rounded up to 3, while 3.1 is rounded up to 4
- Let x represent the call length in minutes; we want to know ***the smallest integer that is larger or equal to*** x
- Mathematically, we are computing

$$\lceil x \rceil$$

this is called
' the **ceiling** of x '

Phone Bill – Steps


BEGIN
Read minutes from user
Compute rounded_minutes = $\lceil minutes \rceil$
Compute charge $= 2 \times rounded\_minutes$
Report charge on screen
END

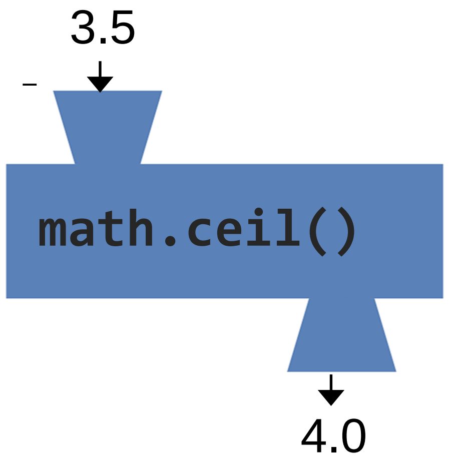
www.fppt.info

Phone Bill – Program

```
1: import math
2: minutes_str = input('Enter call length in minutes: ')
3: minutes = float(minutes_str)
4: rounded_minutes = math.ceil(minutes)
5: charge = 2*rounded_minutes
6: print(f'Charge is {charge:.2f} Pesos.')
```

- `import math`imports the *math module* that contains additional mathematical functions
- Line 3, the expression is evaluated to $\lceil minute$

```
math.ceil(minutes)
```



Math Module

- In addition to built-in functions, Python provides many mathematical functions and constants in the ***math*** module
- Some common functions and constants are:

Expression	Evaluated to	Remark
`math.ceil(x)`	$\lceil x \rceil$	compute smallest integer larger or equal to x
`math.floor(x)`	$\lfloor x \rfloor$	compute largest integer smaller or equal to x
`math.cos(x)`	cos(x)	compute cosine of angle x in radians
`math.sin(x)`	sin(x)	compute sine of angle x in radians
`math.degrees(x)`	$180x/\pi$	convert angle x from radians to degrees
`math.radians(x)`	$x\pi/180$	convert angle x from degrees to radians
`math.sqrt(x)`	$\sqrt{x}$	compute square-root of x
`math.pi`	π	yield the value of π (approx.3.14159)
`math.e`	e	yield the value of e (approx.2.71828)

Math Functions: Examples

```
>>> import math
>>> math.fabs(-12.34)
12.34
>>> math.ceil(3.29)
4
>>> math.floor(3.29)
3
>>> math.cos(math.pi/4)
0.7071067811865476
>>> math.pow(5,3)
125.0
>>> math.sqrt(2)
1.4142135623730951
```

```
>>> import math
>>> math.exp(1)
2.718281828459045
>>> math.log(4)
1.3862943611198906
>>> math.log10(100)
2.0
>>> math.log(8,2)
3.0
>>> math.pi
3.141592653589793
>>> math.e
2.718281828459045
```

www.fppt.info

Two ways of importing

- Importing a module as a whole
  - Names inside are accessed via the module name

```
import math
value = math.cos(math.pi/2)
```

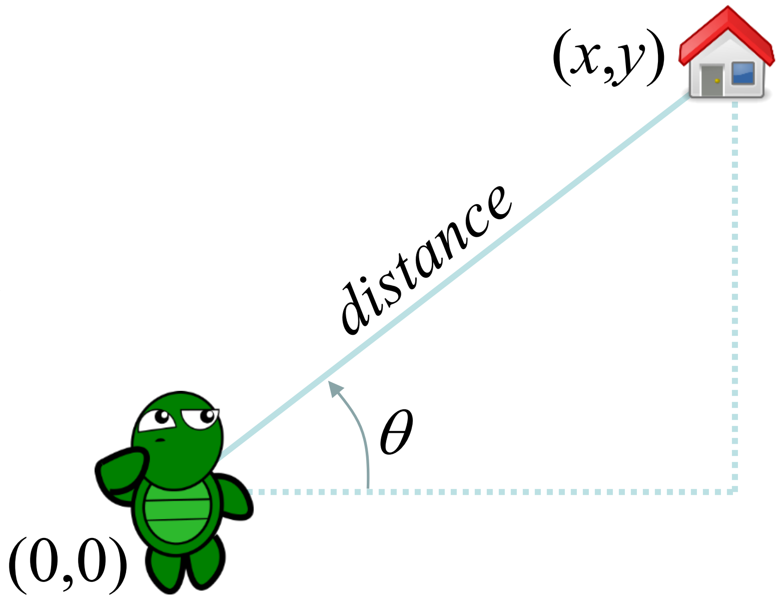
- Importing only specified names inside a module
  - These imported names can be used directly

```
from math import cos, pi
value = cos(pi/2)
```

www.fppt.info

Task: Bring Turtle Home

- Our little robotic turtle is lost in the field. Please help guide him from his location at (0,0) to his home at (*x*,*y*)
- He cannot walk very fast, so we must head him to the right direction so that he can walk with the shortest distance
- Write a program to take the values *x* and *y*, then report the values of θ and *distance*



Bring Turtle Home - Ideas

- Again, we need to analyze the relationship among all the variables to solve the two unknowns
- From Pythagorean theorem

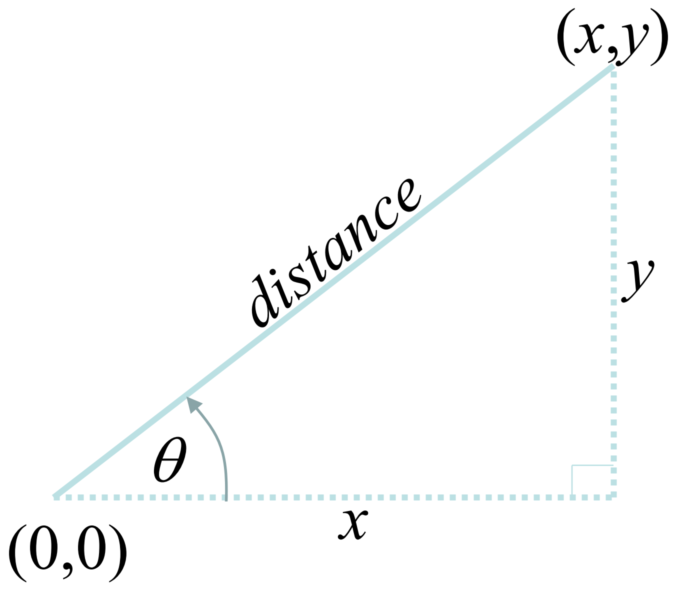
$$distance^2 = x^2 + y^2$$

- And from Trigonometry

$$\tan\theta = \frac{y}{x}$$

- Therefore,

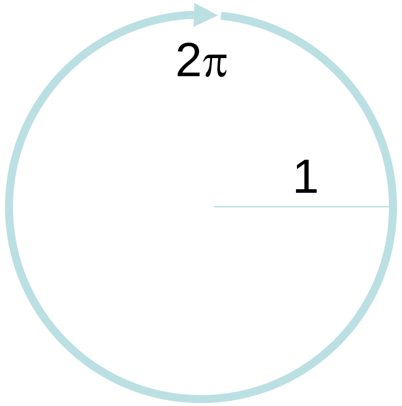
$$distance = \sqrt{x^2 + y^2} \quad \text{and} \quad \theta = \arctan\frac{y}{x}$$



www.fppt.info

Caveats – Radians vs. Degrees

- In most programming languages, the unit of angles used by trigonometry functions is ***radians***, not degrees
- A full circle, 360 degrees, is 2π radians
- In Python, we can use `math.radians()` and `math.degrees()` to convert between radians and degrees



```
>>> math.degrees(math.asin(1))
90.0
```

Bring Turtle Home – Program

```
import math

x = float(input('Enter x: '))
y = float(input('Enter y: '))
distance = math.sqrt((x*x) + (y*y))
heading = math.degrees(math.atan(y/x))
print(f'Heading: {heading:.2f} degree')
print(f'Distance: {distance:.2f} units')
```

Composition

- Some functions *return* a value, which can be used as part of an expression and/or an argument of another function
- As part of an expression:

```
rounded_minutes = math.ceil(minutes)
charge = 2*rounded_minutes
```

↓

```
charge = 2*math.ceil(minutes)
```

www.fppt.info

Composition

- Function that has a value can also be part of an argument of another function:

```
minutes_str = input('Enter call length in minutes: ')
minutes = float(minutes_str)
```

↓

```
minutes = float(input('Enter call length in minutes: '))
```

From now on, we will write input statement this way when reading a number

www.fppt.info

Task: Savings Account

- When you have a savings account, the bank usually deposits interest back into your account every year
- You would like to know how much money you will have after a certain number of years
- Write a program that
  - lets user input the principal, rate (%), and years
  - outputs the amount you will have after the specified number of years

www.fppt.info

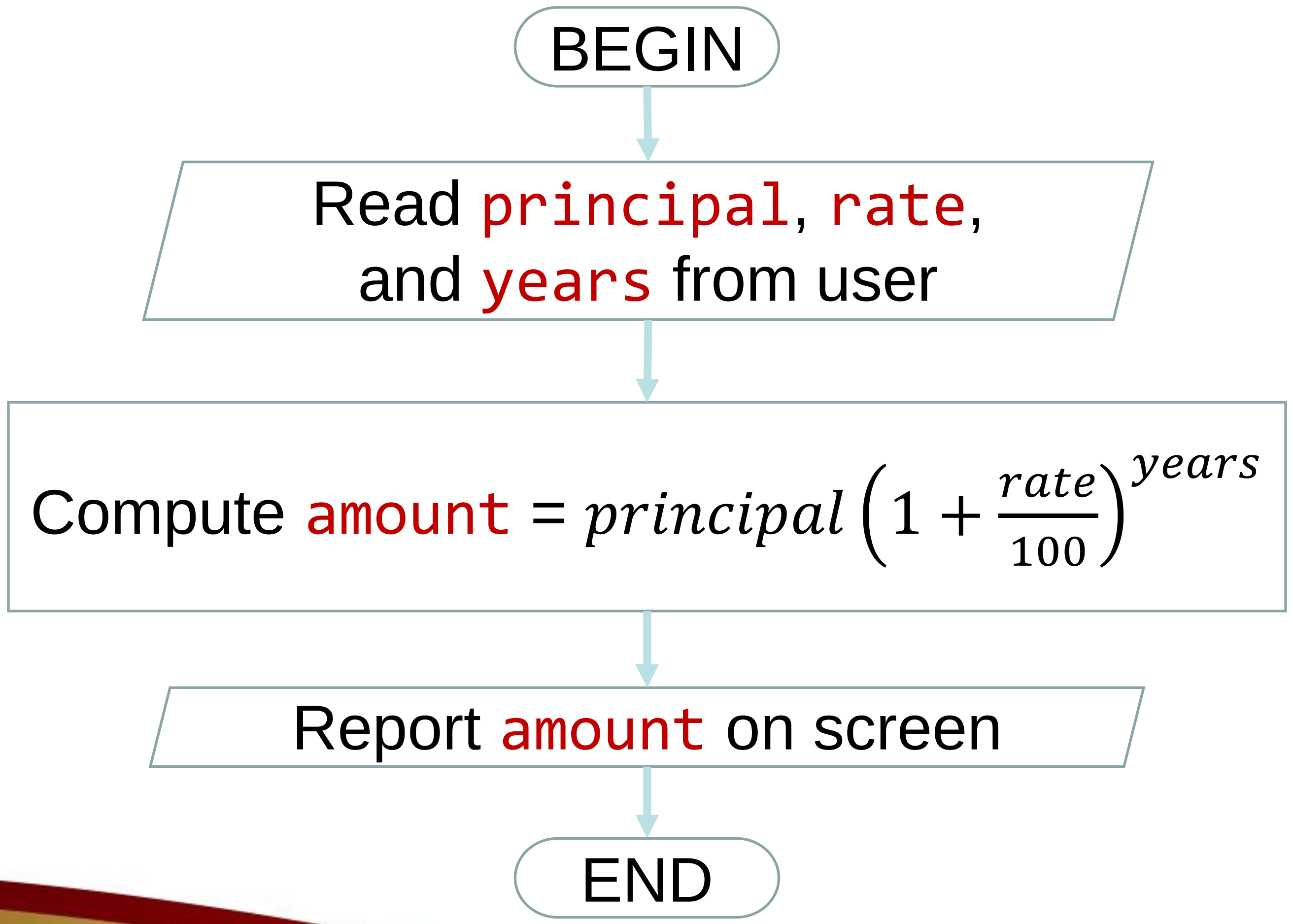
Savings Account - Ideas

- Let us analyze the relationship among the amount in the account, principal (*p*), rate (*r*), and years (*n*)

Year	Amount
0	p
1	$p\left(1+\frac{r}{100}\right)$
2	$p\left(1+\frac{r}{100}\right)\left(1+\frac{r}{100}\right)=p\left(1+\frac{r}{100}\right)^2$
3	$p\left(1+\frac{r}{100}\right)^2\left(1+\frac{r}{100}\right)=p\left(1+\frac{r}{100}\right)^3$
:	:

- It follows that on nth year, the amount will be $p\left(1+\frac{r}{100}\right)^n$

Savings Account – Steps


BEGIN
Read principal, rate, and years from user
Compute amount = $principal\left(1+\frac{rate}{100}\right)^{years}$
Report amount on screen
END

www.fppt.info

Savings Account – Program

```
import math

principal = float(input('Principal (Baht): '))
rate = float(input('Rate (% per year): '))
years = int(input('Time (years): '))
amount = principal * math.pow(1 + rate/100, years)
print(f'Amount: {amount:.2f}')
```

Same as:

```
(1 + rate/100)**years
```

www.fppt.info

Defining your own function

- Python lets you *use* and also *define* functions
- We *group* up a number of statements/computations and then we *give them a name*
  - This enables reuse of these statements (just like we use built-in functions)
- Using your own functions, program is:
  - shorter by eliminating repetitive codes
  - easier to understand
  - less error-prone
- If a change is needed, make it in one place

A Simple Function Definition

- Functions must be defined before use

def is the keyword that means: I am **def**ining a function

Name of your defined function: follow the naming rules

```python
def my_hello(name):
    print(f'Hello, {name}.')

my_hello('Jon Snow')
```

Parameters of your defined function (as many as you need)

Statement in your defined function

Your program that calls your defined function

www.fppt.info

Function Declaration Syntax

Function Name

0 or more parameter names

header

```
def function_name(...):
```

body

```
        ...
        ...
        statements
        ...
```

Very important
Spaces in front of every statement must be the same

www.fppt.info

Task: Circle Area

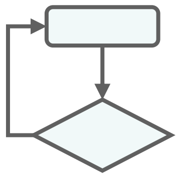
- Program will ask the user to input the radius value of a circle, calculate the circle's area, and then print the resulting circle's area to screen.

www.fppt.info

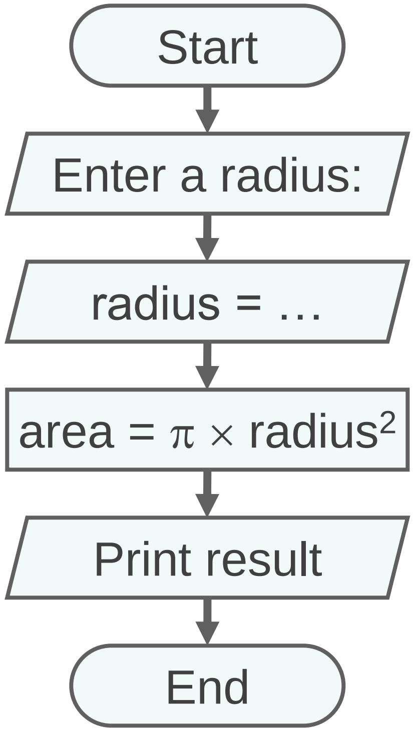
Circle Area - Ideas

- Need to know what is the radius of the underlying circle
- Compute the circle’s area
  - area = $\pi \times$ radius $\times$ radius
- Show the result to screen

Circle Area - Steps

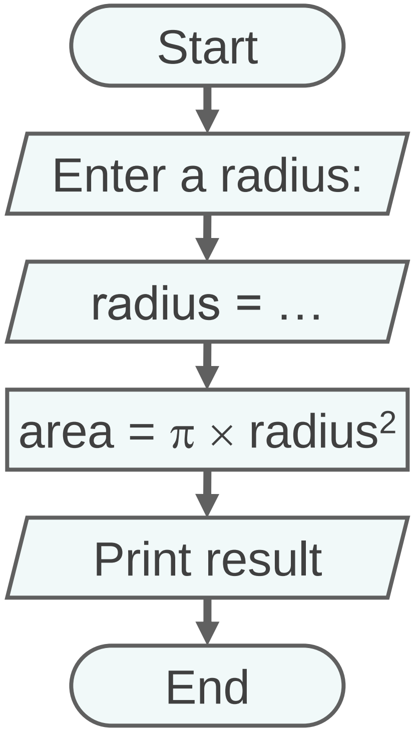


- Tell user to input the radius to the program
- Get input radius from the user
- Calculate the Area
  - area = $\pi \times$ radius $\times$ radius
- Print the resulting Area
- Pause the screen



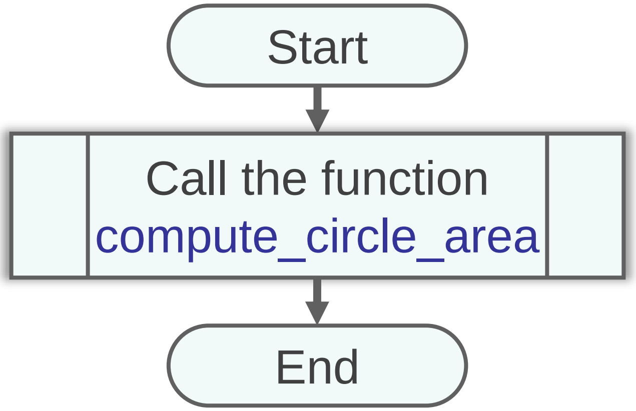
Circle Area – Program#1

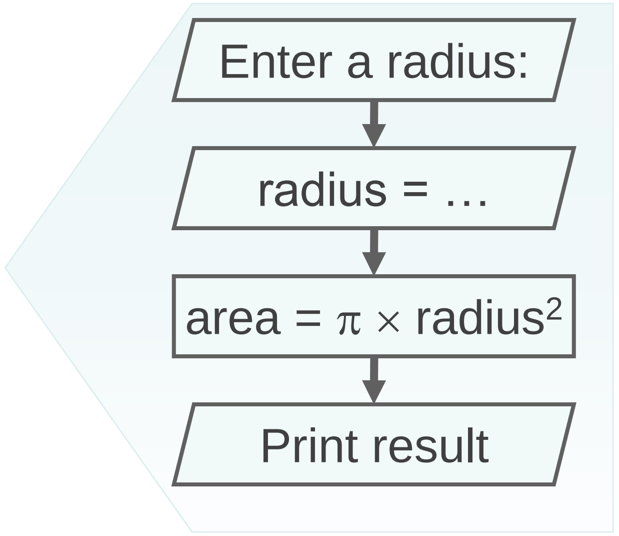
```python
radius = float(input('Enter a radius: '))
area = math.pi*radius**2
print('Area of a circle with radius
         {radius} is {area:.2f}')
```



www.fppt.info

Circle Area – Program#2





```python
def compute_circle_area():
    radius = float(input('Enter a radius: '))
    area = math.pi*radius**2
    print(f'Area of a circle with radius {radius} is {area:.2f}')

compute_circle_area()
```

www.fppt.info

Function Names

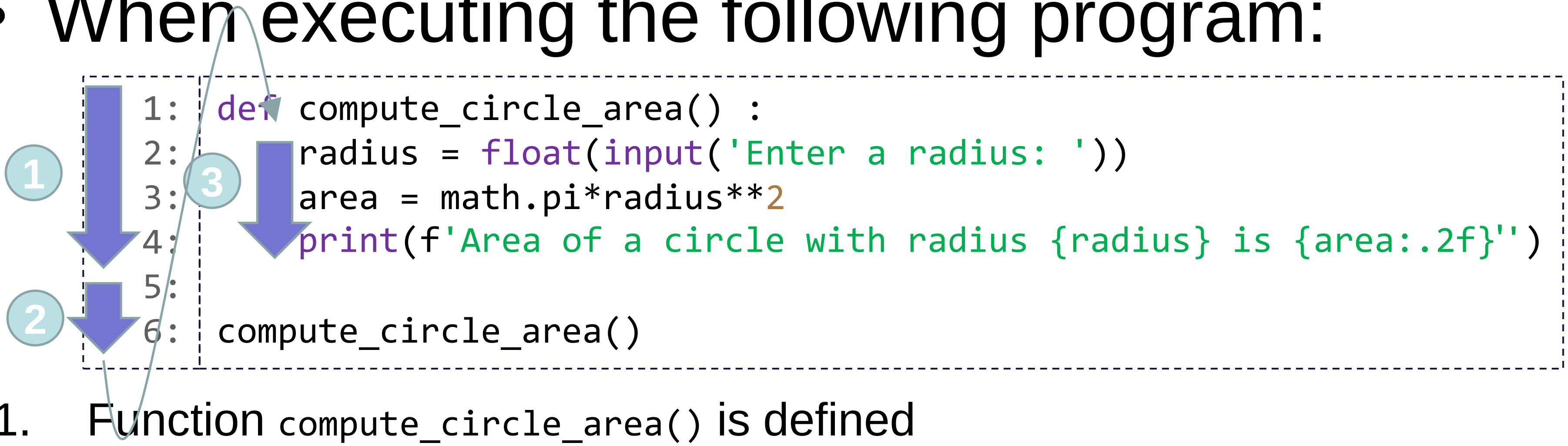
- Follow the conventions for creating an identifier

```
def compute_circle_area():
    ...
}
```

- Examples:
  - calculate_sales_tax()
  - assign_section_number()
  - display_results()
  - convert_input_value()

Flow of Execution

- When executing the following program:



```python
def compute_circle_area() :
    radius = float(input('Enter a radius: '))
    area = math.pi*radius**2
    print(f'Area of a circle with radius {radius} is {area:.2f}')

compute_circle_area()
```

1. Function `compute_circle_area()` is defined
2. Main part of user program—calling `compute_circle_area()`
3. Statements within the function are executed

Returning a Value

- Function can *return* a value
- Recall a built-in function–`float()`

```
celsius = float('35')
```

  - `float('35')` returns a value that equal to 35.0
  - In other words, the term `float('35')` is equal to 35.0
- You can write your own function that returns a value
  - Requires `return` keyword

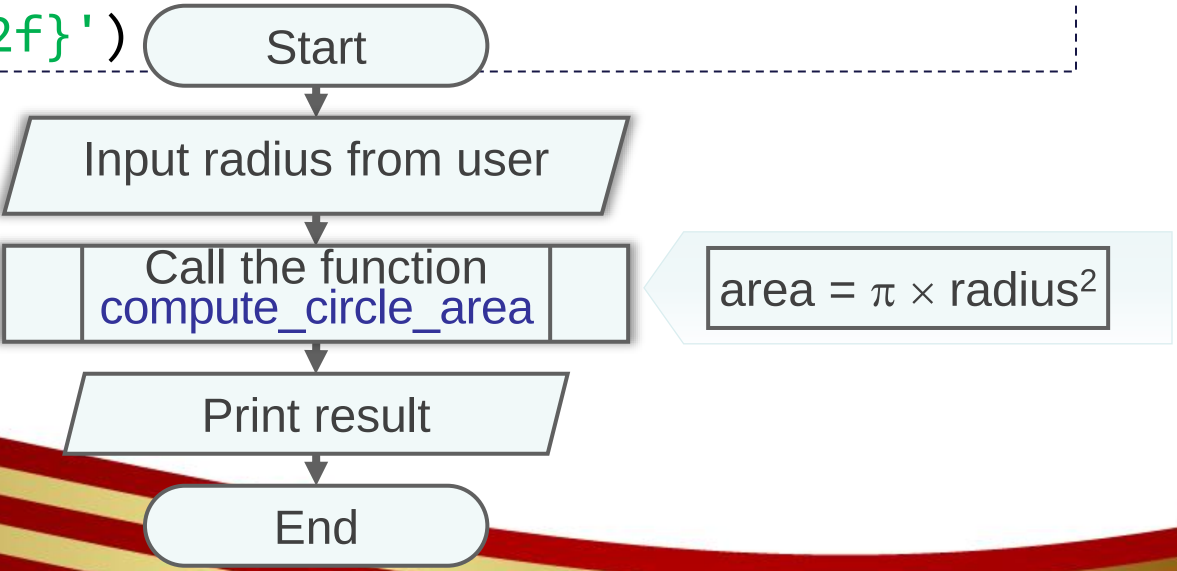
www.fppt.info

Circle Area – Program#3

```
def compute_circle_area(radius):
    circle_area = math.pi*radius**2
    return circle_area

r = float(input('Enter a radius: '))
area = compute_circle_area(r)
print('Area of a circle with radius {r} is
        {area:.2f}')
```

Value in `circle_area` is ***returned***

Notice how the `return` keyword is used



www.fppt.info

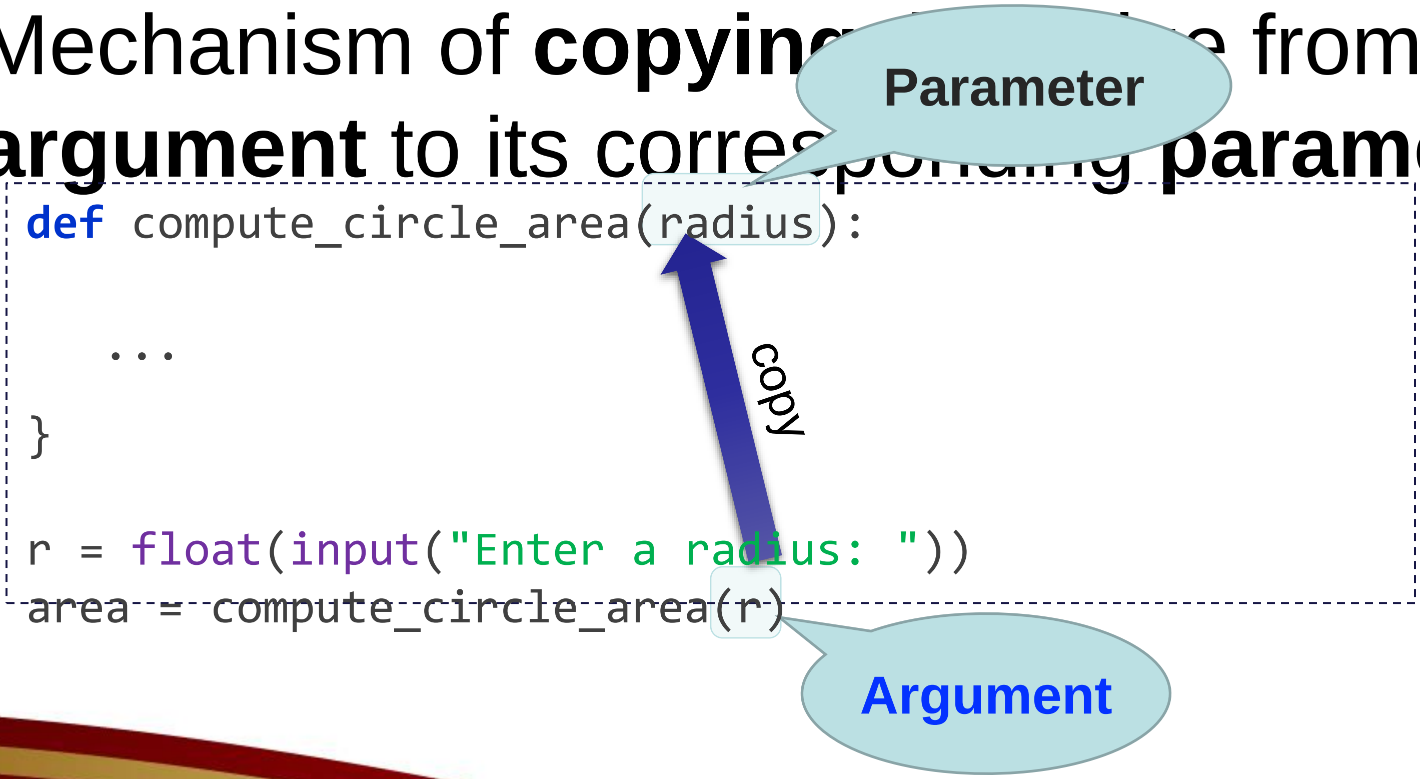
Circle Area – Program#4

```python
def compute_circle_area(radius):
    return math.pi*radius**2

r = float(input('Enter a radius: '))
area = compute_circle_area(r)
print('Area of a circle with radius {r} is
         {area:.2f}')
```

www.fppt.info

Passing Parameters

- Mechanism of **copyin** e from an **argument** to its corres ponding **parameter**

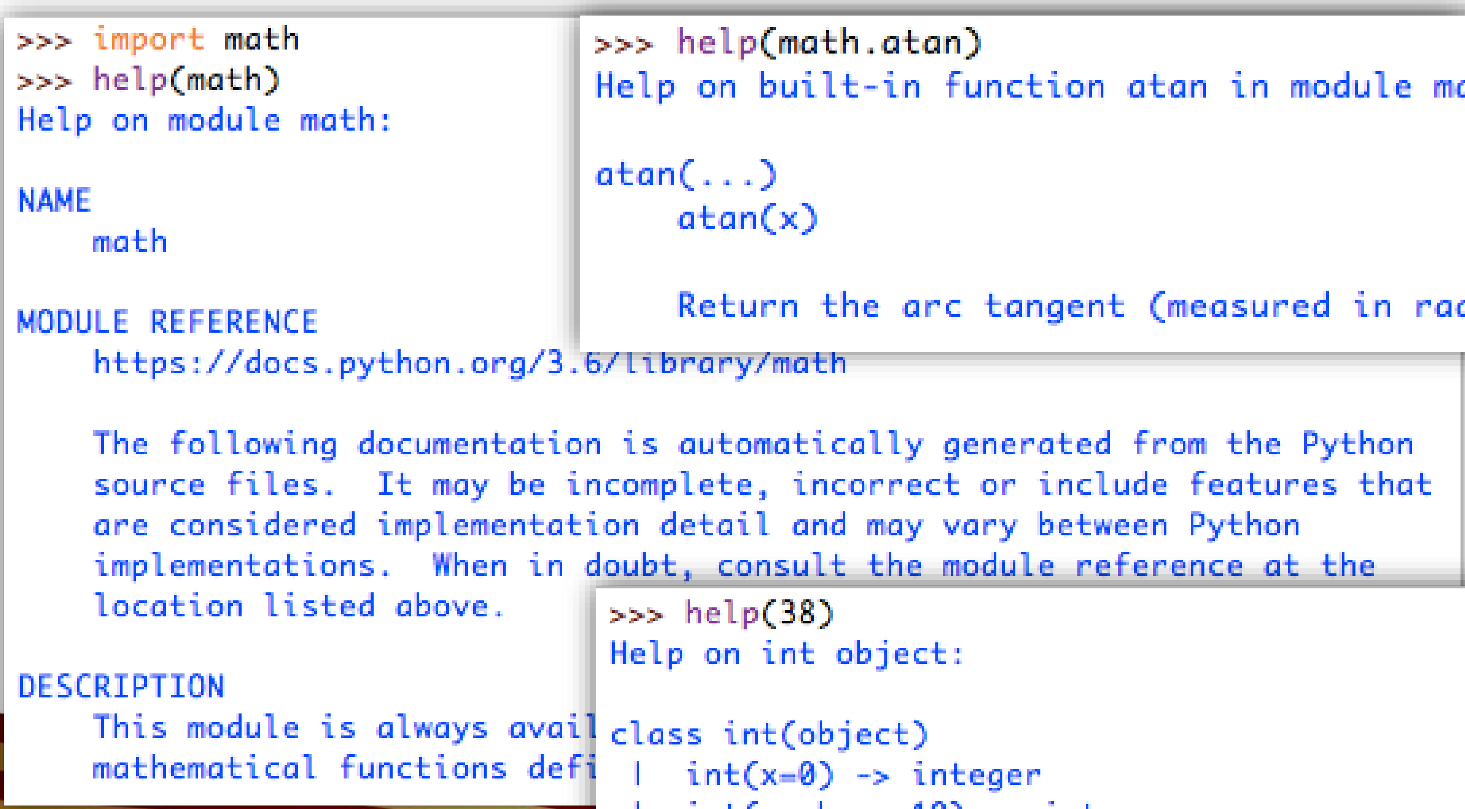


```
def compute_circle_area(radius):

    ...
}

r = float(input("Enter a radius: "))
area = compute_circle_area(r)
```

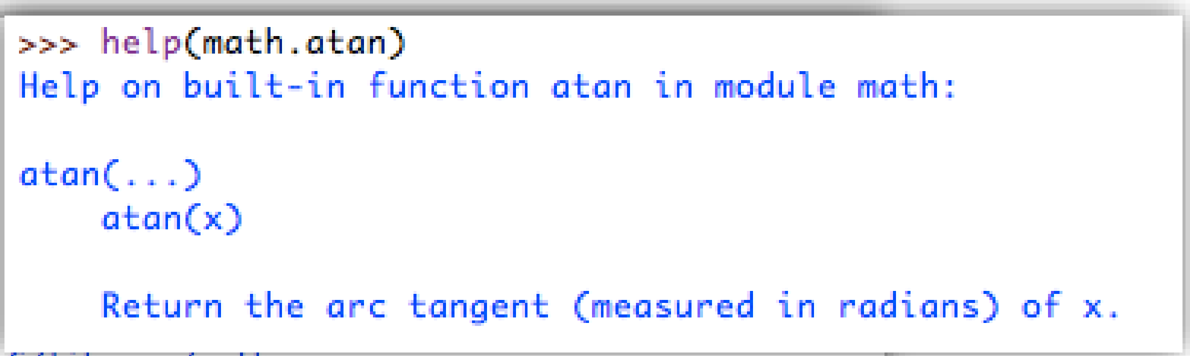
www.fppt.info

Seeking help

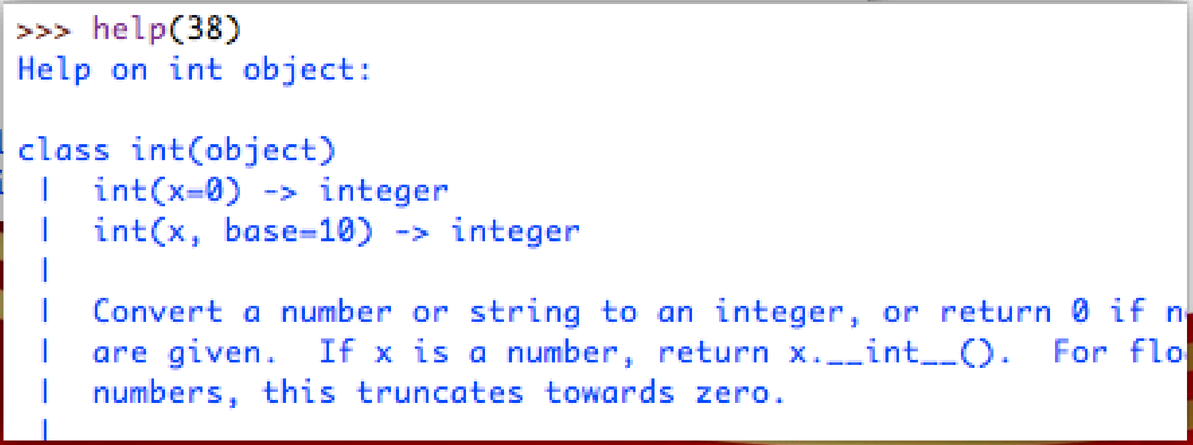
- The built-in `help()` function can be used in the shell to provide more details about any object



```
>>> import math
>>> help(math)
Help on module math:

NAME
    math

MODULE REFERENCE
    https://docs.python.org/3.6/library/math

    The following documentation is automatically generated from the Python
    source files.  It may be incomplete, incorrect or include features that
    are considered implementation detail and may vary between Python
    implementations.  When in doubt, consult the module reference at the
    location listed above.

DESCRIPTION
    This module is always avail
    mathematical functions defi
```



```
>>> help(math.atan)
Help on built-in function atan in module math:

atan(...)
    atan(x)

    Return the arc tangent (measured in radians) of x.
```



```
>>> help(38)
Help on int object:

class int(object)
 |  int(x=0) -> integer
 |  int(x, base=10) -> integer
 |
 |  Convert a number or string to an integer, or return 0 if n
 |  are given.  If x is a number, return x.__int__().  For flo
 |  numbers, this truncates towards zero.
 |
```

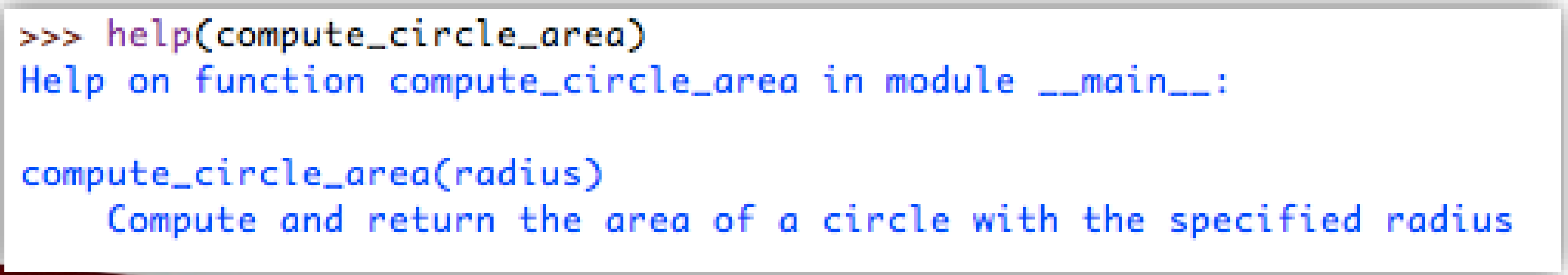
www.fppt.info

Providing help with docstring

- A docstring can be added at the beginning of a function to be shown by the `help()` function

```python
def compute_circle_area(radius):
    """
    Compute and return the area of a circle
    with the specified radius
    """
    return math.pi*radius**2
```

docstring



```
>>> help(compute_circle_area)
Help on function compute_circle_area in module __main__:

compute_circle_area(radius)
    Compute and return the area of a circle with the specified radius
```

Conclusion

- Subroutine/function is a group of statements
- There are built-in functions and user-defined functions
- Python provides many useful mathematical functions in its math module

Syntax Summary

- Read a number from keyboard

```
variable_name = int(input(str))
variable_name = float(input(str))
```

- Define a function

```
def function_name(parameters...):
      ...
      ...
      return value
```

optional

optional

www.fppt.info

Syntax Summary

- Import a module as a whole

```
import module_name
```

- Import some objects from a module

```
from module_name import obj1, obj2
```

References

- Python's math module
  https://docs.python.org/3/library/math.html
- Python docstring
  https://www.python.org/dev/peps/pep-0257/

www.fppt.info