Hacettepe University

Computer Engineering Department

Programming in python

BBM103 Introduction to Programming Lab 1

Week 3

Fall 2016

Writing Your First Program

- Example 1: **printing output**

```
print('Hello, World!')
```

New function:
`print()`

- Example 2:

```
language = "Python programming language"
print(language)
```

- Example 3: **printing multiple lines**

```
print("""
|===========BBM103===========|
|                            |
|  Welcome to Programming!   |
|  Python is easy and fun!   |
|                            |
|============================|
""")
```

- Example 4: **customizing the separator between printed items**

```
print("L", "i", "n", "u", "x", sep=".")

print(*"Linux", sep=".")
```

  - **Output of both lines:** `L.i.n.u.x`

Taking Input

- Example 5: **taking the input as a string**

```
name = input("What is your name? ")
print("Hello", name, end="!\n")
```

New function:
`input()`

- Example 6: **converting the input to integer**

```
number = int(input("Please enter a number: "))
print("The square of the number: ", number ** 2)

# We can do the same operation with pow() function: pow(number,2)
```

New function:
`int()`

- Example 7:

```
number1 = int(input("Enter the first number: "))
number2 = int(input("Enter the second number: "))
print(number1, "+", number2, "=", number1 + number2)
```

- Example 8: **formatting output**

New function:
`str.format()`

```
url = input("Please enter the url")
print("Error! Google Chrome couldn’t find {} ".format(url))
```

- Example 9: **formatting output**

```
print('{0} and {1}'.format('Tom', 'Jerry'))
```

  - **Output:** `Tom and Jerry`

File Input/Output

If we want, we can write output to a file instead of screen.

- Example 10: **writing output to a file**

```
output = open("newFile.txt","w")
print("Hello World",file=output)
output.close()
```

Open a new file for writing

Write to the file

Close the file

!

This is important! You will use this method in your assignments and quizzes.

Control Flow - Branching

The simplest branching statement is a **conditional**. `<condition>` has a value `True` or `False`. `<expression>` is evaluated if `<condition>` is `True`.

1

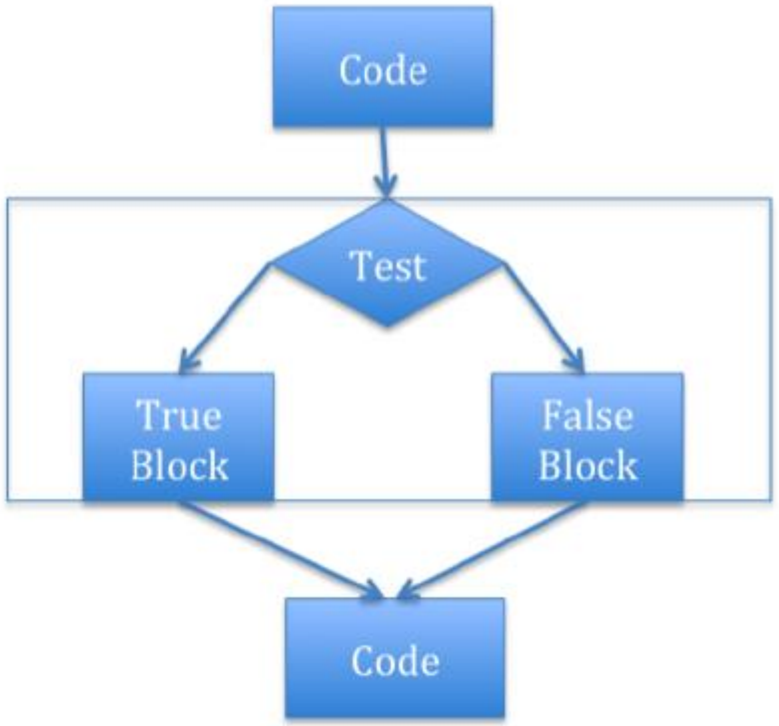
```
if <condition>:
    <expression>
    <expression>
    ...
```

2

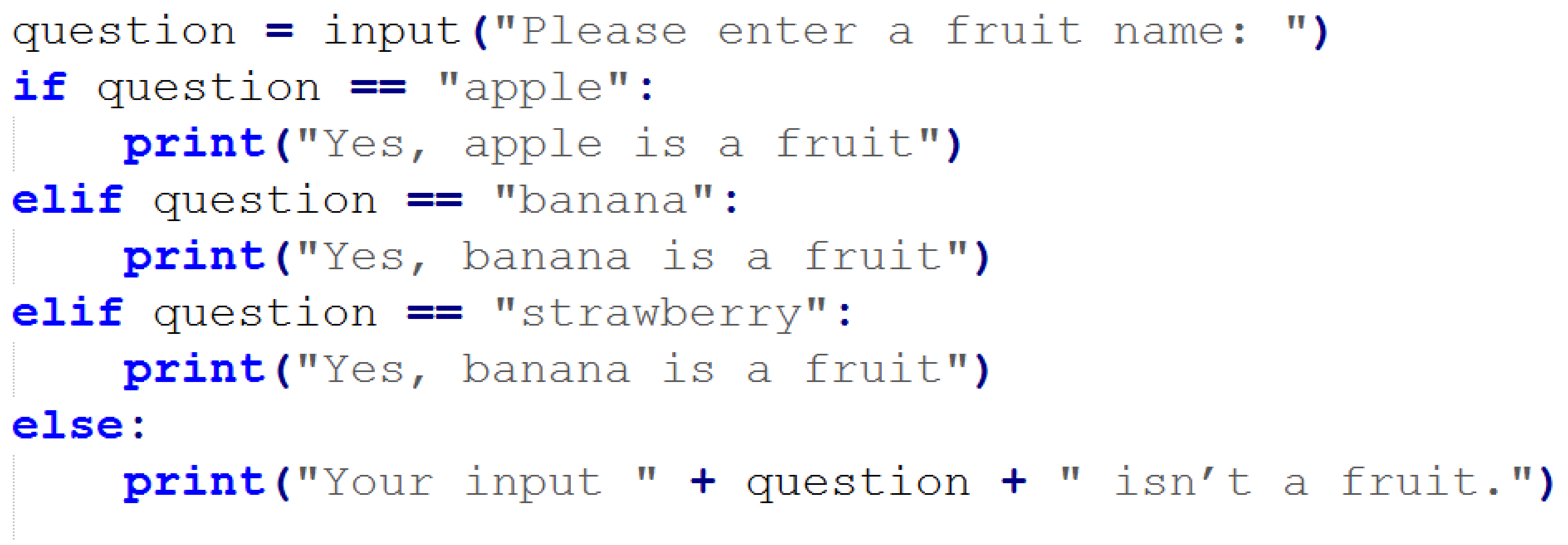
```
if <condition>:
    <expression>
    <expression>
    ...
else:
    <expression>
    <expression>
    ...
```

3

```
if <condition>:
    <expression>
    <expression>
    ...
elif <condition>:
    <expression>
    <expression>
    ...
else:
    <expression>
    <expression>
    ...
```



- Example 11: **control flow**



```python
question = input("Please enter a fruit name: ")
if question == "apple":
    print("Yes, apple is a fruit")
elif question == "banana":
    print("Yes, banana is a fruit")
elif question == "strawberry":
    print("Yes, banana is a fruit")
else:
    print("Your input " + question + " isn’t a fruit.")
```

‘==‘ is a comparison operator as opposed to ‘=‘ sign which is an assignment operator.

- Comparison operators in Python:

Operator	Description	Example
==	If the values of two operands are equal, then the condition becomes true.	(a == b) is not true.
!=	If values of two operands are not equal, then condition becomes true.	
<>	If values of two operands are not equal, then condition becomes true.	(a <> b) is true. This is similar to != operator.
>	If the value of left operand is greater than the value of right operand, then condition becomes true.	(a > b) is not true.
<	If the value of left operand is less than the value of right operand, then condition becomes true.	(a < b) is true.
>=	If the value of left operand is greater than or equal to the value of right operand, then condition becomes true.	(a >= b) is not true.
<=	If the value of left operand is less than or equal to the value of right operand, then condition becomes true.	(a <= b) is true.

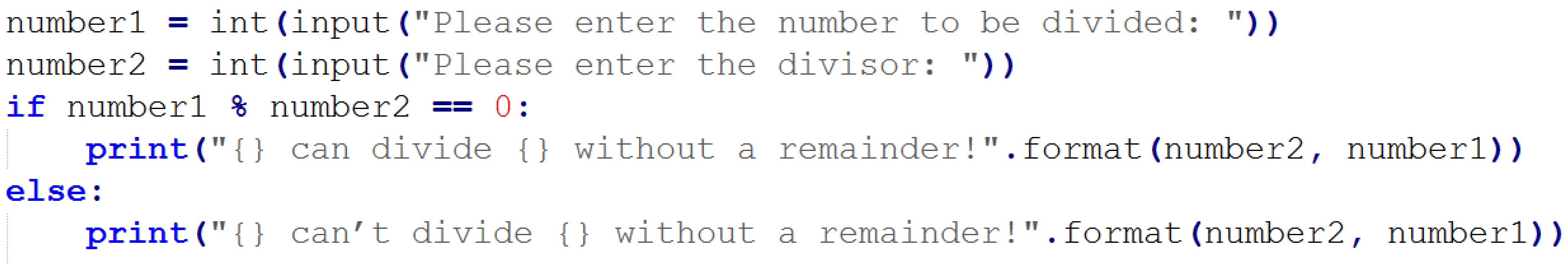
- Example 12: **control flow continued**

New function:
`len()`

```python
username = input("Your username: ")
password = input("Your password : ")
total_weight = len(username) + len(password)
message = "Your username and password has a total of {} characters!"
print(message.format(total_weight))
if total_weight > 40:
    print("The total length of your username and password ",
          "should not exceed 40 characters!")
else:
    print("Welcome to the system!")
```

len() returns the length of its argument.

- Example 13: **control flow continued – checking if two numbers are divisible**



```python
number1 = int(input("Please enter the number to be divided: "))
number2 = int(input("Please enter the divisor: "))
if number1 % number2 == 0:
    print("{} can divide {} without a remainder!".format(number2, number1))
else:
    print("{} can’t divide {} without a remainder!".format(number2, number1))
```

‘%‘ is a modulus operator which divides the left hand operand by the right hand operand and **returns the remainder**.

Control Flow – For Loops

```
for <variable> in range(some_number):
    <expression>
    <expression>
    ...
```

- Each time through the loop, `<variable>` takes a new value. It starts with the smallest value, and in the next loop it gets incremented, and so on, until it reaches the final value in the specified range.

- Example 14: **printing numbers in a given range**

```
for i in range(10):
    print(i)
```

New function:
`range()`

- Example 15: **printing numbers grater than a specified value**

```
numbers = "123456789"
for i in numbers:
    if int(i) > 3:
        print(i)
```

`range(number)` generates integers from 0 up to, but not including `number`.

- Example 16: **printing characters that are not in a string**

```python
first_text = "This is a sample text for testing."
second_text = "This is another sample text."
for letter in first_text:
    if letter not in second_text:
        print(letter)
```

- Example 17: **printing numbers divisible by three**

```python
for number in range(2,50):
    if int(number) % 3 == 0:
        print(number)
```

`range(start, stop)` **generates integers from** `start` **up to** `stop`**, but not including** `stop`**.**

- Example 18: **finding the cube root**

```
x = int(input('Enter an integer: '))
answer = None
cube_root_found = False
for i in range(0, abs(x)+1):
    if i**3 == abs(x):
        answer = i
        cube_root_found = True
if not cube_root_found:
    print (x, 'is not a perfect cube')
else:
    if x < 0:
        answer = -answer
    print('Cube root of', x,'is', answer)
```

New function:
`abs()`

This is not a very efficient algorithm, but it gets the job done!
Food for thought:
- **Why?**
- **How can we make it more efficient?**

`abs(number)` **returns the absolute value of** `number`.

Functions

Defining functions:

```
def function_name(arguments):
    function_body
    ...
```

Calling functions:

```
function_name(arguments)
```

- Example 19: **defining a <u>void</u> function (function that does not return a value)**

```
def greeting(name):
    print("Good afternoon, " + name + ".")

greeting("Emre")
```

Defining function

Calling function

- **Output:**

```
Good afternoon, Emre.
```

- Example 20: **defining a <u>fruitful</u> function (function that returns a value)**

```
def maximum(x, y):
    if x > y:
        return x
    else:
        return y
max_number = maximum(1, 5)
print("The maximum of two numbers is", max_number)
```

The function **returns** a value

- Example 21: **Calculating area of plane shapes**

```
def triangle_area(b, h):
    return b*h/2
def square_area(a):
    return a*a
def rectangle_area(a, b):
    return a*b
user_choice = int(input("""Choose a shape you wish to calculate the area of:
(1) Triangle
(2) Square
(3) Rectangle\n1-3: """))
if user_choice == 1:
    base = int(input("Enter the length of the triangle base: "))
    height = int(input("Enter the height of the triangle: "))
    print("The area of the triangle is", triangle_area(base, height))
elif user_choice == 2:
    side = int(input("Enter the length of the square side: "))
    print("The area of the square is", square_area(side))
elif user_choice == 3:
    width = int(input("Enter the width of the rectangle: "))
    height = int(input("Enter the height of the rectangle: "))
    print("The area of the triangle is", rectangle_area(width, height))
else:
    print("Sorry, there is no such option.")
```

`\n` is the newline character

Things to remember

- Indentation is very important in Python! To indicate a block of code in Python, you **must indent each line of the block by the same amount**.

- **Practice makes perfect**: the more you practice programming the easier it gets. It is easy to get stuck in the beginning, but don’t get discouraged. Work with simple examples first. Move on to the harder examples when you have fully grasped the simple ones.

- It is a lot of fun telling your computer what to do! **Stay motivated**.