Hillview International School: Year 9

Programming in Python:
Nested if statements and casting

Lesson outcomes

By the end of this lesson, you will be able to:

- Understand casting: changing a variable from one type to another
  - Casting “input” values
- Understand why, when and how to use nested if statements
  - The affect of indentation
- Be aware of common terms in programming and their meanings

Casting

- What is it?
  - Variables come in lots of different types:
    - Strings, Integers, Floating point numbers, etc
  - Sometimes you need to change values into different types
  - Casting is performed by a series of functions
- When do you need to do it?
  - The `input()` function always returns a String – you often need to change the return type
  - The `print()` function always prints Strings – you often need to change variables

Casting inputs and outputs

- The return type from the **`input()`** function is always a String
- You can change the return types by casting:
  - **`whole_number = int(input("Enter a whole number: "))`**
  - **`float_number = float(input("Enter a floating point number: "))`**

- The **`input()`** function always takes Strings:
- You can change the types by casting:
  - **`print(str(whole_number), "is an integer")`**
  - **`print(str(float_number), "is a floating point number")`**

Activity 1: The quiz of 9

- Create a program that asks questions and prints their results, eg:
  - What is 9 divided by 2?
  - You answered 4.5!
- Use the following questions:
  - What is 9 divided by 2?
  - What is 9 multiplied by 2?
  - How do you spell 9?

You have 10 minutes!

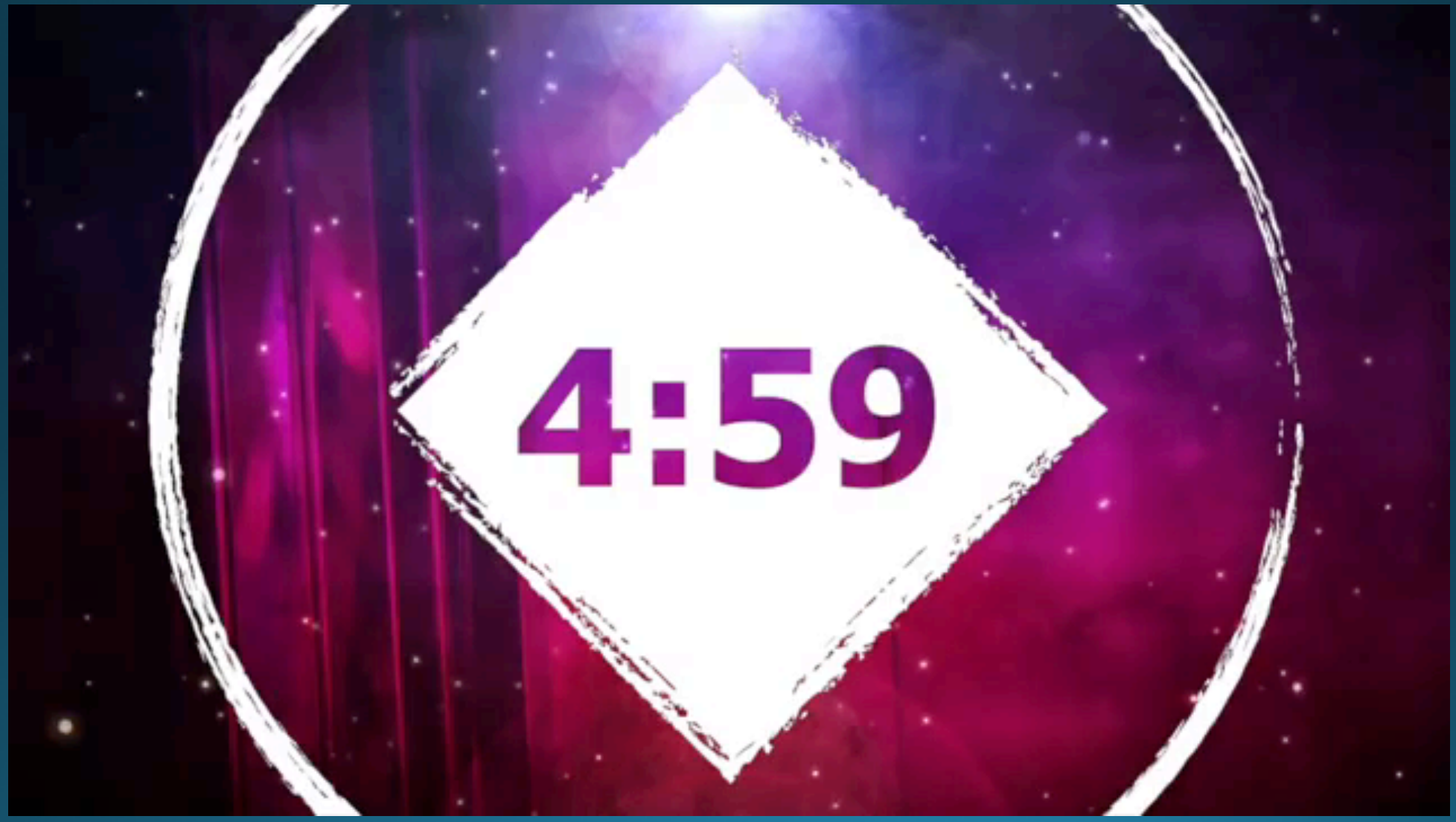
Recapping if / elif / else

```
if <CONDITION IS TRUE>:
    # Do this when condition is true

elif <ALTERNATIVE CONDITION IS TRUE>:
    # Do this when alternative condition is true

else:
    # Do this if none of the clauses is true
```

Activity 2: Positive or negative number?

```
def test_number(number):
    if number == 0:
        print(str(number), “is zero”)
    elif number > 0:
        print(str(number), “is a positive number”)
    else:
        print(str(number), “is a negative number”)

number = float(input(“Enter a number: ”))
test_number(number)
```

You have 5 minutes!



Nested if statements

- Sometimes you only want to test an if-statement when another if-statement is true. This is when you use a nested if.

```
if <CONDITION1 IS TRUE>:
        # Do this when condition 1 is true

    if <CONDITION2 IS TRUE>:
        # Do this when condition 1 and 2 are true
    else:
        # Do this if condition 1 is true and condition 2 is false
else:
        # Do this if condition 1 is false (regardless of condition 2)
```

- ***Notice the indentation***

Activity 3: Logon system

```
def test_logon(username, password):
    if username == “admin”:
        if password == “hillview”:
            print(“Password correct!”)
        # End password test
    # End username test
# End test_logon function

username = input(“Enter username: ”)
password = input(“Enter password: ”)
test_logon(username, password)
```

You have 10 minutes!

Terminology

- Type - the form that the data is held in, eg, String, int, float
- String - a piece of text
- Variable - a named storage location for data
- Statement - an instruction to the computer
- Condition - a true/false test, often in if-statements
- Clause - another word for condition

- Why is program not spelt “programme”?

Activity 4 – match the terms to definitions

Lesson summary

You should now be able to:

- Understand casting: changing a variable from one type to another
  - Casting “input” values
- Understand why, when and how to use nested if statements
  - The affect of indentation
- Be aware of common terms in programming and their meanings

On Monday:

- Loops
  - For loops
  - While loops