Functions and Classes

Functions are an example of decomposition

```
def welcomeMessage(name):
  print("Welcome", name)

myName = "Bob"
welcomeMessage(myName)
```

Classes are like blueprints for creating objects in OOP

```
class myClass:
  def __init__(self, teacher, room):
    self.teacher = teacher
    self.room = room

class1 = myClass("Mrs. T", "Room 7B")
```

if/else

```
if myMarks >= 40:
  print("You passed!")
elif myMarks < 40:
  print("You failed :( ")
else:
  print("Your marks aren't ready yet.")
```

Useful Operations

File Handling

- `myFile = open("textfile.txt", "r")`
- `myFile.read()`
- `myFile.write(paragraph)`
- `myFile.close()`

String Handling

- `print("Hello \nworld!")`
- `print(len("CompSci"))`
- `print("Capital".upper())`
- `print("Th1s w1ll b3 f4l5e".isdigit())`
- `print("**RECEIPT**".strip("**"))`

Exception Handling

```
try:
     print("Try something")
except:
     print("Do this if it fails")
```

List Operations

- `myList.append("item")`
- `myList.pop()`
- `print(myList[0])`

Data Types

Mutable	Immutable
List `myList = ["pear", 1]`	String `"CS!", "2022"`
Set `mySet = {"hello", "bye", 2.9}`	Integer `5, 101, 5678`
Dictionary `myDict = {'a': 1, 'b': 2, 'c': 3}`	Float `3.1, 78.9, 100.0`
	Tuple `x = (2)`
	Boolean `true, false`

*Mutable → **can** be modified after creation*
*Immutable → **cannot** be modified after creation*

python™ Cheatsheet

Loops

For loops

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

or

```
myList = ["Leicester", "CS"]
for i in list:
  print(i)
```

While loops

```
while 0 < 1:
  print("Never ending loop!")
```

Operators

Arithmetic: `+, -, *, /, **, //, %`

Comparison: `<, >, <=, >=, ==, !=`

Logical: `and, or, not`

Assignment: `=, +=, -=` etc.

Leicester CompSci Soc

Data Types

Primitive	Reference
int `5, 101, 5678`	String `String myString = "hello"`
short `-32,768… 32,767`	array `Integer[] intArray = {1,2,3}`
long -2^{63}… $2^{63} - 1$	class `class myClass()`
float `3.1, 78.9, 100.0`	
boolean `true, false`	
char `'a', 'z', '3'`	

Primitive data types – immutable

Reference data types – mutable (except String)

Operators

Arithmetic: `+, -, *, /, %, ++` (increment by 1), `--` (decrement by 1)

Comparison: `<, >, <=, >=, ==, !=`

Logical: `&& (and), || (or), ! (not)`

Assignment: `=, +=, -=` etc.

Uses OOP paradigm

Loops

For loops

```
for (int i=0; i<5; ++i) {
  System.out.println("Counter = " + i);
}
```

While loops

```
while (true) {
  System.out.println("Loop!");
}
```

Input

Create a Scanner to use:

```
import java.util.Scanner
class Input {
  public static void main(String[] args) {
    Scanner input = new Scanner (System.in);
    System.out.print("Enter a string: ");
    String str = input.nextLine();
    }
}
```

Java Cheatsheet

Functions and Classes

accessModifier = public, private or protected

Creating a function:

```
accessModifier returnType funcName(parameters){
  code;
}
```

Creating a class:

```
accessModifier class className(parameters) {
  attributes - varType varName;
}
```

Creating an object:

```
className objectName = new className(parameters);
```

Output

- `System.out.print();` prints string inside “”
- `System.out.println();` prints string inside “” & moves to beginning of next line in console
- `System.out.printf();` provides string formatting

Leicester CompSci Soc