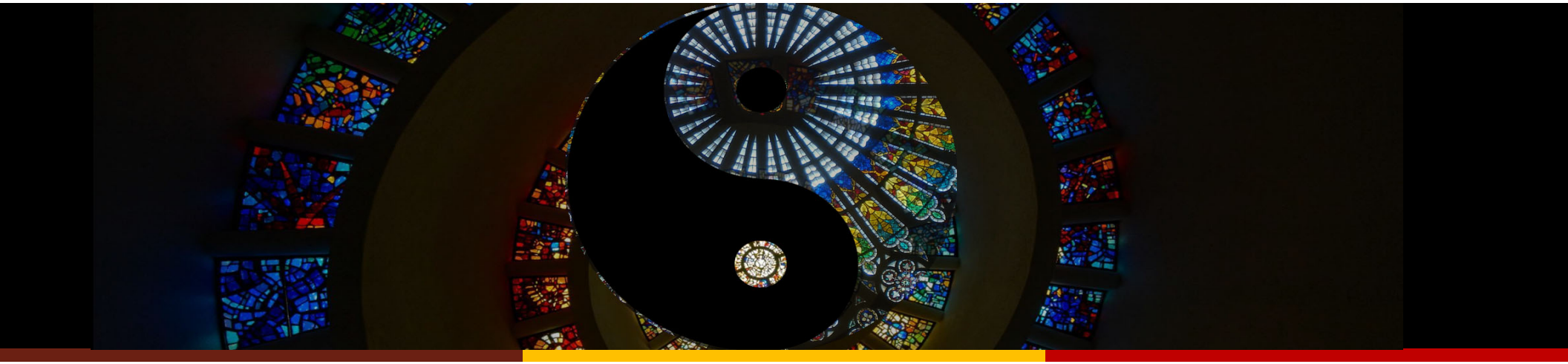


Python For Loop

Logical Thinking of Informatics

Lab 3

Data Operations

+ - ÷ × ...

Data operations – Integers, Floats

- Operations
  - + - ÷×
  - Exponentiation **
  - Remainder %

String Operations - Split

- Split string according to delimiter string
- Return list of substrings
- Example:

```
myString="hello_every_one_!_this_is_a great afternoon"
splitResult=myString.split("_")
print(splitResult)
```

String Operations - Slice

- [x:y]: from x position and before y position
- Example:

```
sliceString=myString[7:12]
print(sliceString)
```

String Formatting Using %s

- %s for string
- %d for signed decimal integer
- %f for floating point real number
- Example

```
tempInt=21
TAName="Debby"
print("%d students are posing questions to GEC 1506 TA %s" % (tempInt,TAName))
```

Iterations

while and for

While Loop

```
inputList=[]
while True:
    try:
        inputList.append(input())
    except EOFError as e:
        break

print(len(inputList))
print(inputList)
```

condition: true
(this is not a very good writing style)

While Loop

```
Line 1:  inputList=[]
Line 2:  copyList=['apple','water','Joe','Jessie']
Line 3:  count=0
Line 4:  while count<3:
Line 5:    try:
Line 6:      inputList.append(copyList[count])
Line 7:      count=count+1
Line 8:    except EOFError as e:
Line 9:      break
Line 10: print(inputList)
```

Syntax of While()

Initialization

While (Condition):

    Action1

    Action 2…n (optional)

    Change condition

Two Types of Loop

- While loop
  - The number of iterations is not known in advance
  - Run as log as the specified condition remains true
  - It should consist of a condition in the loop

- For loop
  - The number of iterations is known beforehand
  - Has definite iteration count
  - Consist of a initialization, condition, update

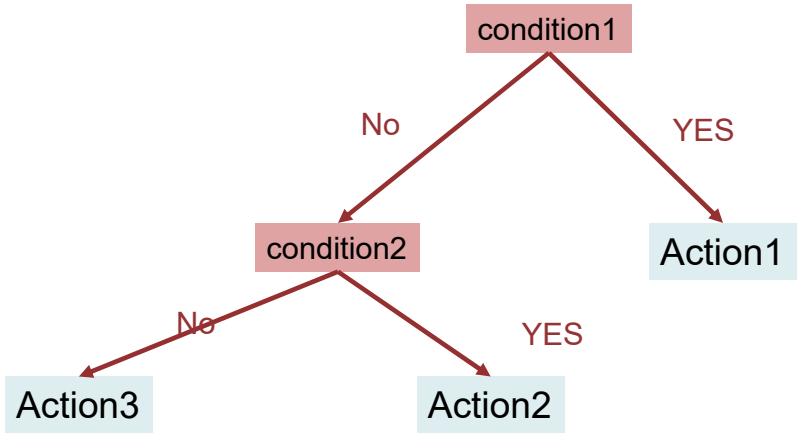
For Loop

- A control flow statement for specifying iteration
  - Allows code to be executed repeatedly

```
inputList=[]
copyList=['apple','water','Joe','Mary','Kevin','Debby']
for count in range (0,5):
  try:
    inputList.append(copyList[count])
  except EOFError as e:
    break
print(len(inputList))
print(inputList)
```

Condition

Syntax of IF




```
if (condition1):
    # Code to execute → Action1
elif (condition2):  # optional
    # Code to execute if condition1 is false and condition2 is true  →Action2
else:   #optional
    # Code to execute if both condition1 and condition2 are false   → Action3
```

Example for IF

- We need to specify such condition, such as:

```
MyMood=input()
if (MyMood=="Happy"):
   print("Go to GEC1506 classes and Study Hard!")
elif (MyMood=="Sad"):
   print("Skip the classes and stay in the bed")
else:
   print("Go to GEC1506 classes and do nothing")
```

Another Example for IF

- We need to specify such condition, such as:

```
MyMood=input()
if (MyMood=="Angry"):
    print("Shout to my Dog")
else:
    print("Do homework!")
```

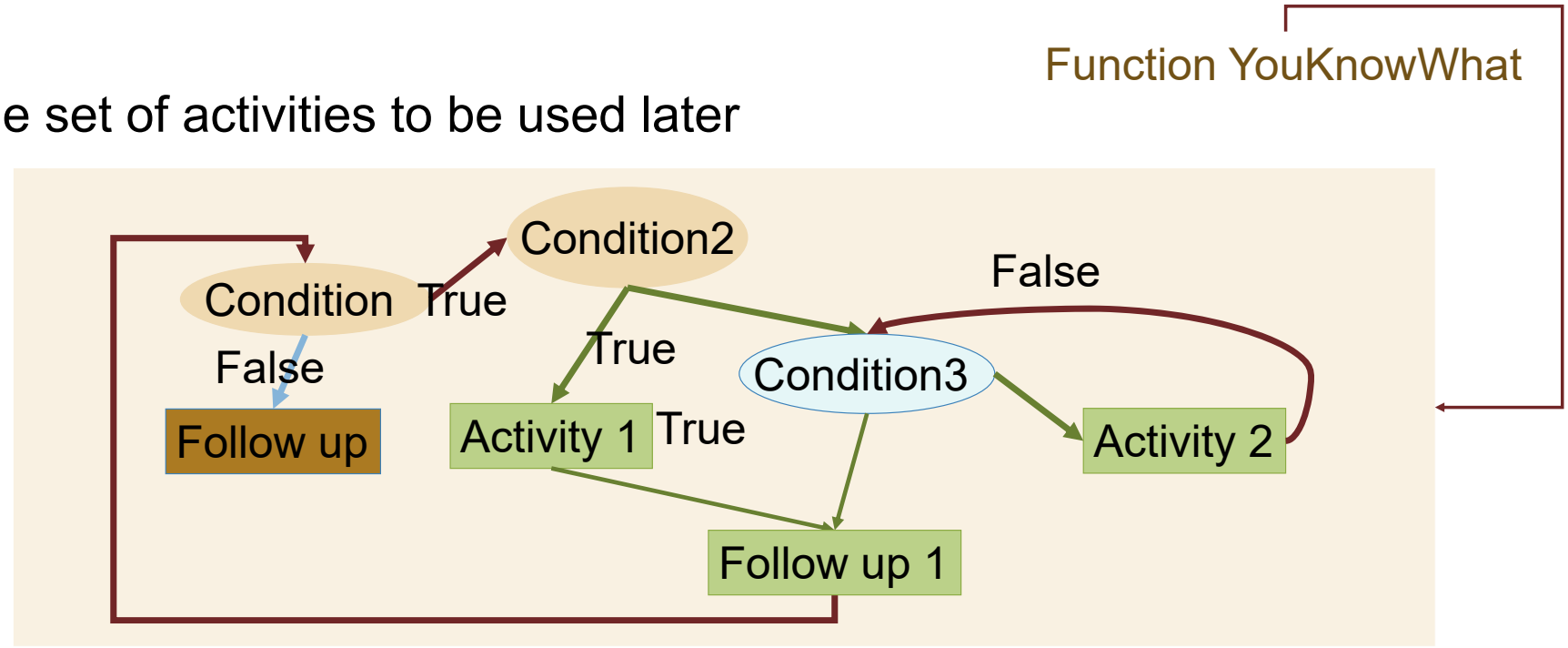
Let's Practice

```
Line 1: from datetime import date
Line 2: today=date.today()
Line 3: if (today=date(2024,4,8)):
Line 4:   print("we will have Lab3 class")
Line 5: else:
Line 6:   print("No Lab Class!")
```

Function

Function

- The set of activities to be used later



- Can be called for many times
  - With different inputs

Try This Code

```
Line 1: skip=2
Line 2: boundary=[0,100]
Line 3: myList=[]
Line 4: for value in range(boundary[0],boundary[1],skip):
Line 5:     if ((value%4)! =0):
Line 6:         myList.append(value)
Line 7: print(myList)
```

Compare

```
Line 1: skip=2
Line 2: boundary=[0,100]
Line 3: myList=[]
Line 4: for value in range(boundary[0],boundary[1],skip):
Line 5:    if ((value%4)! =0):
Line 6:       myList.append(value)
Line 7: print(myList)
```

```
Row 1: def genIntList(low,high,skip, exclude):
Row 2:   myList=[]
Row 3:   for value in range(low,high,skip):
Row 4:     if ((value%exclude)!=0):
Row 5:       myList.append(value)
Row 6:   return(myList)
```

- def: define a function
- genIntList: a function name
- (low, high, skip, exclude): input variables
- return(myList): return the output

How to Use a Function?

```
Row 1: def genIntList(low,high,skip, exclude):
Row 2:   myList=[]
Row 3:   for value in range(low,high,skip):
Row 4:     if ((value%exclude)!=0):
Row 5:       myList.append(value)
Row 6:   return(myList)
```

- Follow the definition of function
- For example:

```
outList=genIntList(5,105,3,5)
print(outList)
```

- Output:

Advantages of Using Functions

- Modularity: Break down complex tasks into smaller one
- Readability: Easier to trace the codes
- Reusability: Can be called multiple times
- Abstraction: Abstract the details
- Organization: Help to structure the codes
- Simplify Testing and Debugging