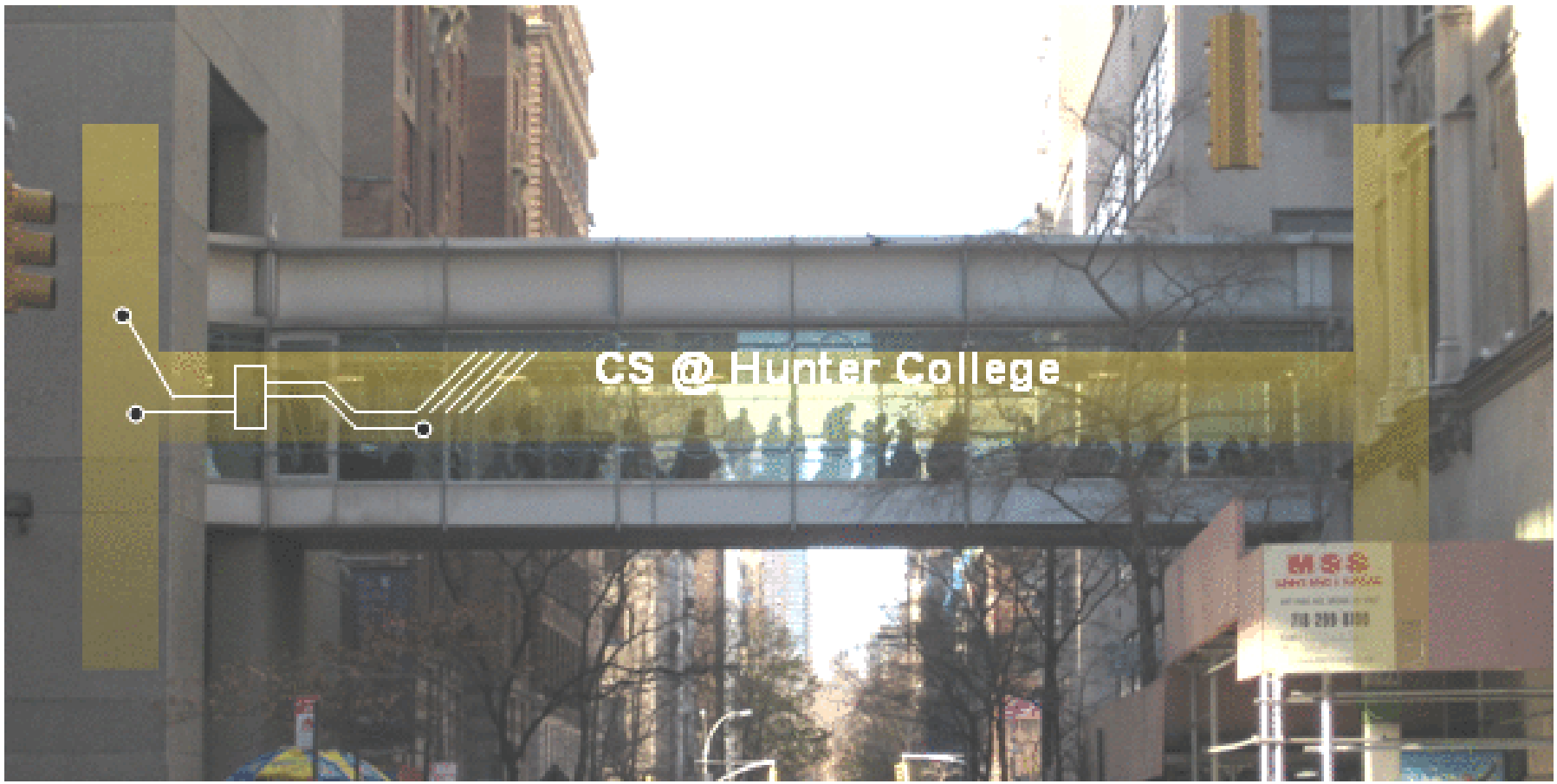
CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

Frequently Asked Questions

From lecture slips & recitation sections.

- Where is the final? When are we taking it?
  Tuesday, 22 May, 9-11am, 118 North.
- Can we do more on colors, images, numpy & matplotlib?
  Yes, we will in Labs 4, 6-9 & Lectures 6-9.
  Today, we'll focus on decisions, and logical expressions & circuits.
- What is pseudocode? Why do we use it?
  Pseudocode is the "informal high-level description of the operating principle of a computer program or other algorithm."
  We use it to write down the ideas, before getting deep into the details.
- What was that % symbol? Why is that math?
  It's the symbol for remainder (or modulus). Ex: `11 % 5` *is* `1`.
- What are types of variables?
  Different kinds of information takes different amounts of space.
  Types we have seen so far: `int`, `float`, `str` *and objects (e.g. turtles).*
- How can I tell strings from variables?
  Strings are surrounded by quotes (either single or double).
  Variables names (identifiers) for memory locations are not. Ex: `'num'` *vs.* `num`.

Today's Topics

- Recap: Indexing, Slicing, & Decisions
- Logical Expressions
- Circuits

Lecture Slip: In Pairs or Triples...

Some review:

1
```
motto = "Mihi cura futuri"
print(motto[2:4])
print(motto[2:4].upper())
```

2
```
ER = "The future belongs to those who believe in the beauty of their dreams."
print(ER.upper()[2], ER[13], ER[2], "a", ER[15], ER[14], "r  R.")
```

3
```
import turtle

tess = turtle.Turtle()
myWin = turtle.Screen()     #The graphics window
commands = input("Please enter a command string: ")

for ch in commands:
    #perform action indicated by the character
    if ch == 'F':           #move forward
        tess.forward(50)
    elif ch == 'L':         #turn left
        tess.left(90)
    elif ch == 'R':         #turn right
        tess.right(90)
    elif ch == '^':         #lift pen
        tess.penup()
    elif ch == 'v':         #lower pen
        tess.pendown()
    elif ch == 'B':         #go backwards
        tess.backward(50)
    elif ch == 'r':         #turn red
        tess.color("red")
    elif ch == 'g':         #turn green
        tess.color("green")
    elif ch == 'b':         #turn blue
        tess.color("blue")
    else:                   #for any other character
        print("Error: do not know the command:", c)
```

Recap: Indexing & Slicing

```
motto = "Mihi cura futuri"
print(motto[2:4])
print(motto[2:4].upper())
```

Recap: Indexing & Slicing

```
motto = "Mihi cura futuri"
print(motto[2:4])
print(motto[2:4].upper())
```

M	i	**h**	**i**		c	u	r	a		f	u	t	u	r	i
0	1	**2**	**3**	4	5	6	7	8	9	10	11	12	13	14	15

Output:

```
hi
HI
```

Recap: Indexing & Slicing

```
ER = "The future belongs to those who believe in the beauty of their dreams."
print(ER.upper()[2], ER[13], ER[2], "a", ER[15], ER[14], "r  R.")
```

T	h	**e**		f	u	t	u	r	e		b	e	**l**	**o**	**n**	g	s
0	1	**2**	3	4	5	6	7	8	9	10	11	12	**13**	**14**	**15**	16	17

Output:

E l e a n o r R.

Python Tutor

```
import turtle

tess = turtle.Turtle()
myWin = turtle.Screen()     #The graphics window
commands = input("Please enter a command string: ")

for ch in commands:
    #perform action indicated by the character
    if ch == 'F':           #move forward
        tess.forward(50)
    elif ch == 'L':         #turn left
        tess.left(90)
    elif ch == 'R':         #turn right
        tess.right(90)
    elif ch == '^':         #lift pen
        tess.penup()
    elif ch == 'v':         #lower pen
        tess.pendown()
    elif ch == 'B':         #go backwards
        tess.backward(50)
    elif ch == 'r':         #turn red
        tess.color("red")
    elif ch == 'g':         #turn green
        tess.color("green")
    elif ch == 'b':         #turn blue
        tess.color("blue")
    else:                   #for any other character
        print("Error: do not know the command:", c)
```

(Demo with `pythonTutor`)

In Pairs or Triples...

Some challenges with types & decisions:

```
#What are the types:
y1 = 2017
y2 = "2018"
print(type(y1))
print(type("y1"))
print(type(2017))
print(type("2017"))
print(type(y2))
print(type(y1/4.0))

x = int(y2) - y1
if x < 0:
    print(y2)
else:
    print(y1)
```

```
cents = 432
dollars = cents // 100
change = cents % 100
if dollars > 0:
    print('$'+str(dollars))
if change > 0:
    quarters = change // 25
    pennies = change % 25
    print(quarters, "quarters")
    print("and", pennies, "pennies")
```

Python Tutor

```
#What are the types:
y1 = 2017
y2 = "2018"
print(type(y1))
print(type("y1"))
print(type(2017))
print(type("2017"))
print(type(y2))
print(type(y1/4.0))

x = int(y2) - y1
if x < 0:
    print(y2)
else:
    print(y1)
```

(Demo with pythonTutor)

Decisions

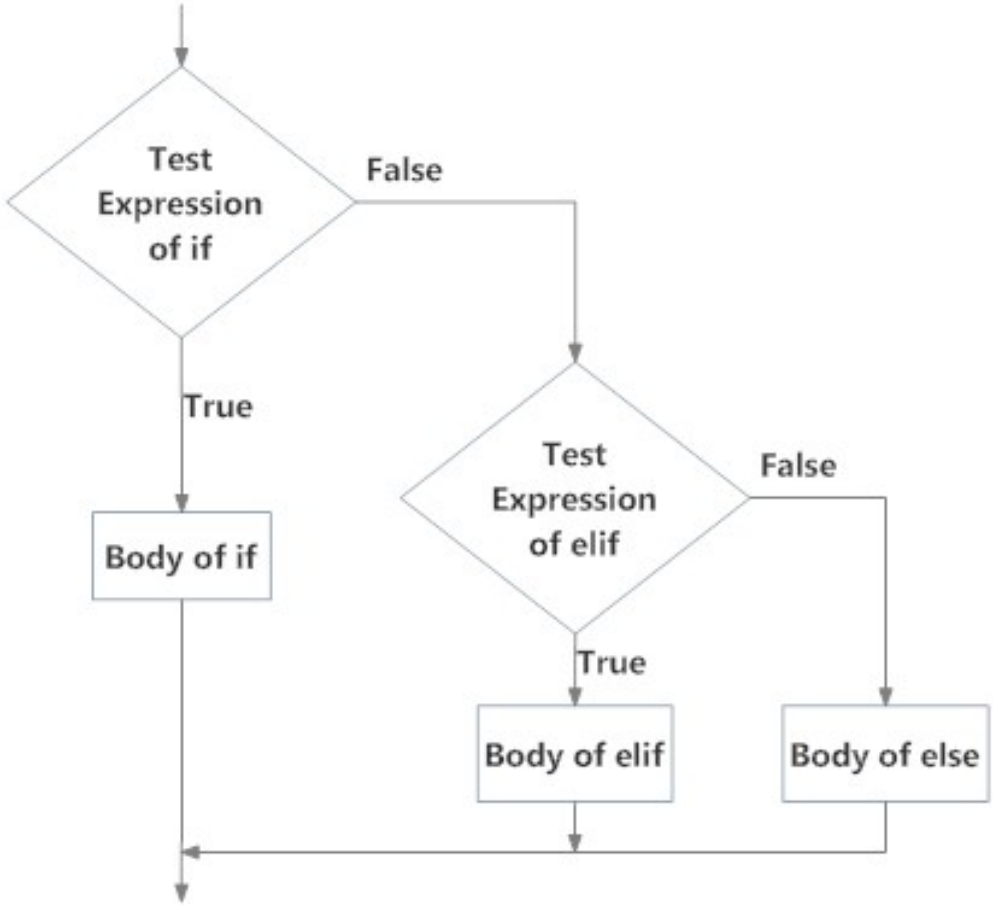
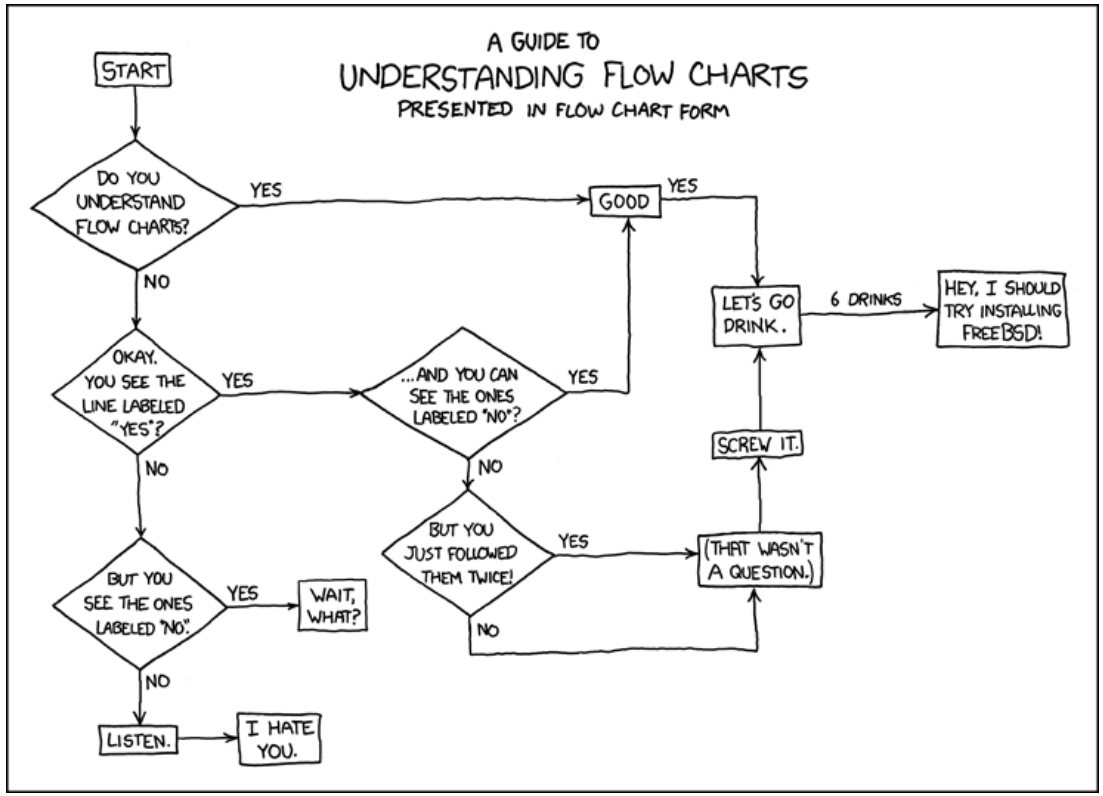


Fig: Operation of if...elif...else statement

(programiz)

Side Note: Reading Flow Charts



(xkcd/518)

In Pairs or Triples

Predict what the code will do:

```
origin = "Indian Ocean"
winds = 100
if (winds > 74):
    print("Major storm, called a ", end="")
    if origin == "Indian Ocean" or origin == "South Pacific":
        print("cyclone.")
    elif origin == "North Pacific":
        print("typhoon.")
    else:
        print("hurricane.")

visibility = 0.2
winds = 40
conditions = "blowing snow"
if (winds > 35) and (visibility < 0.25) and \
     (conditions == "blowing snow" or conditions == "heavy snow"):
    print("Blizzard!")
```

Python Tutor

```
origin = "Indian Ocean"
winds = 100
if (winds > 74):
    print("Major storm, called a ", end="")
    if origin == "Indian Ocean" or origin == "South Pacific":
        print("cyclone.")
    elif origin == "North Pacific":
        print("typhoon.")
    else:
        print("hurricane.")

visibility = 0.2
winds = 40
conditions = "blowing snow"
if (winds > 35) and (visibility < 0.25) and \
     (conditions == "blowing snow" or conditions == "heavy snow"):
    print("Blizzard!")
```

(Demo with `pythonTutor`)

Logical Operators

and

in1		in2	*returns:*
`False`	`and`	`False`	`False`
`False`	`and`	`True`	`False`
`True`	`and`	`False`	`False`
`True`	`and`	`True`	`True`

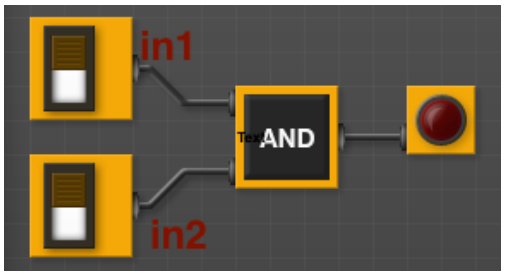
or

in1		in2	*returns:*
`False`	`or`	`False`	`False`
`False`	`or`	`True`	`True`
`True`	`or`	`False`	`True`
`True`	`or`	`True`	`True`

not

	in1	*returns:*
`not`	`False`	`True`
`not`	`True`	`False`

Circuit Demo

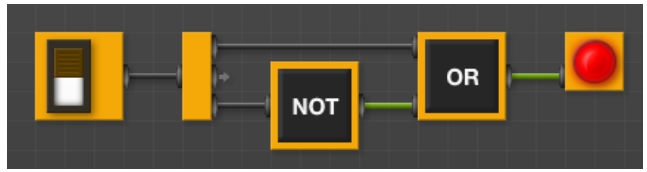


(Demo with `neuroproductions`)

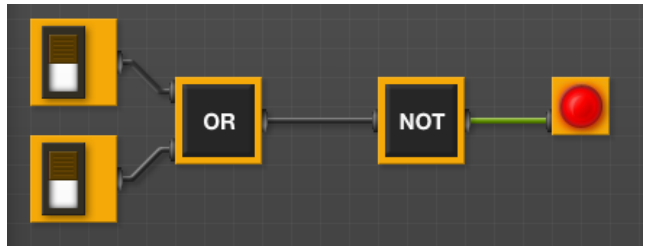
In Pairs or Triples

Predict when these expressions are true:

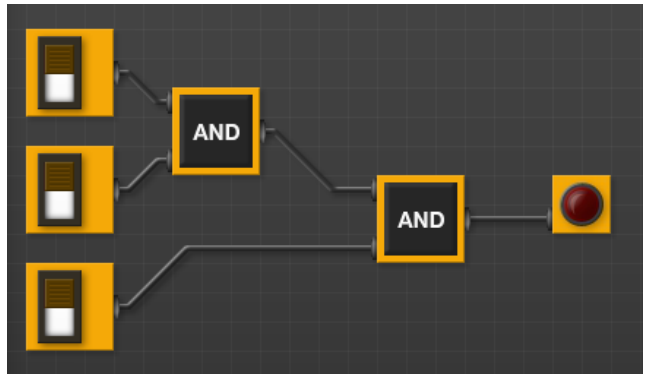
- `in1 or not in1`:



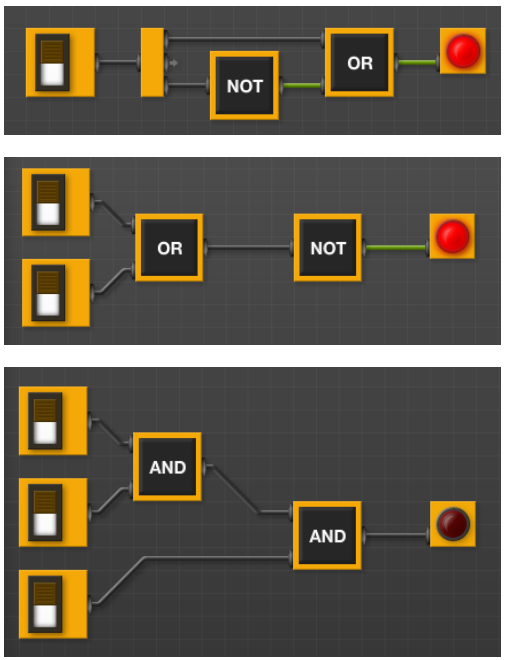
- `not(in1 or in2)`:



- `(in1 and in2) and in3`:



Circuit Demo



(Demo with `neuroproductions`)

Recap

- On lecture slip, write down a topic you wish we had spent more time (and why).
- In Python, we introduced:
  - Decisions
  - Logical Expressions
  - Circuits

Lecture Slip: In Pairs or Triples

From Final Exam, Fall 2017, Version 3:

Name: **EmpID:** CSci 127 Final, V3, F17

1. (a) What will the following Python code print:

```
flist = "speech,worship,want,fear,fdr"
freedoms = flist.split(",")
pres = freedoms[-1]
print(pres.upper())
num = flist.count(",")
print(num, "Freedoms")
for i in range(0,4):
     if i < 2:
          print("\tof", end=" ")
     else:
          print("\tfrom", end=" ")
     print(freedoms[i])
```

Output: