LECTURE 1:

Tuesday, August 13, 2024 9:34 PM

Conditionals....

>	<	==
less than	greater than	equal to
>=	<=	!=
less than or equal to	greter than or equal to	not equal to

We can ask questions with "if".

Again...

```
x = int(input("what's x? "))
y = int(input("what's y? "))
```

boolean expression!

```
if x< y:
....print("x is less than y")
```

no parenthesis, but : used.
also notice indentation

```
if x >y:
....print("x is greater than y")

if x==y:
....print("x is equal to y")
```

The progam checks if each boolen expression is true or false.

- if it decides it's true, it executes the code the expression keeps.
- this example checks every single condition
  - EVEN IF THE FIRST EXPRESSION WAS TRUE

Improving with "else if"...

Again...

```
x = int(input("what's x? "))
y = int(input("what's y? "))

if x< y:
....print("x is less than y")
```

This is the zone of improvement. We only continue checking expressions until one is T.

```
else if x>y:
....print("x is greater than y")

else if x==y:
....print("x is equal to y")
```

In the first exam, we ALWAYS ask a total of 3 question. This only ask 3 on third case.

Again...

```
x = int(input("what's x? "))
y = int(input("what's y? "))

if x< y:
....print("x is less than y")
else if x>y:
....print("x is greater than y")
else:
....print("x is equal to y")
```

Because the final case will ALWAYS be x==y if the first two conditions aren't true, we can safely skip the final comparison and save us some time.

We improved both readability and efficiency.

"Or" keyword.

Allows us to consider two boolean expressions at the same time.

```
if x<y or x>y
....print("x is not equal to y")
else
....print("x is not equal to y")
```

Essentially, we can ask 2 questions and if either are true, execute code.

but let's improve.

```
if x != y
....print("x is not equal to y")
else
....print("x is equal to y").
```

easily inverted

```
if x==y:
....print("x is equal to y")
else
....print("x is not equal to y")
```

* remember, indentation is absolutely a requirement to adhere by.

"and" keyword.

GRADE.PY

```
score = int(input("score: "))
if score >=90 and score<=100
....print("Grade: A")
elif score>=80 and score<90
....print("Grade:B")
            . . . .
else:
....print("Grade:F ")
```

improving by...

```
if 90≤ score ≤100
```

consider knowledge you implicitly know.

```
if score >=90

elif score >=80
    ...
elif score>=60
```

important that this utilizes elif, because we skip the rest of the conditions.

Is a given number even or odd?

% modulo my beloved.

Determines the remainder after division!

reminder...

1 % 3 = 1
2 % 3 = 2
3 % 3 = 0
4 % 3 = 1

use to check even or odd!

```
x = int(input("give x: "))
if x%2 ==0
....print("even")
else:
....print("odd")
```

Implement a function to determine even or odd

```
def is_even(n)
.... if n%2 ==0
........return true
....else:
........return false
```

We return a boolean, a value that can only be true or false.

We cannot pass values by their address.

now use it...

```
if is_even(x):
.... print("x is even")
else:
....print("x is odd")
```

Pythonic Expressions

```
def is_even(n)
.... return True if n%2==0 else False
```

From 4 lines, to a single line.

```
def is_even(n)
....return(n%2 ==0)
```

most succint solution. If this expression is true, it returns as such & vice versa.

"match" keyword

house.py

```
name=input("What's your name? ")
if name =="Harry"
....
if name =="Hermione"
....
if name =="Ron"
....
if name =="Draco"
....
else:
.... Who?
```

improvement!

```
if name=="Harry" or name=="Hermione" or...
```

now....match

```
match name:
.... case 'Harry':
........
.... case "Hermione":
........
.... case "Ron":
........
.... case "Draco":
........
.... case _:
........
```

```
match name:
....case:"Harry"|"Ron"|"Herm"
........
.... case "Draco":
........
.... case_:
........
```

most succint manner.
no break statements needed!