ICS 226
Network and Server-Side Programming

- **X01A**: Please pick a station with a number in the range from **21 - 40** (left side of the room as you come in)
- **X01B, X02A**: Please pick a station with a number in the range from **6 - 20** (right side of the room)
- **X02B**: Please pick a station with a number in the range from **1 - 5 or 41 - 44** (last row on the right or middle)
- Record your name on the schedule attached to the KVM; reserve only times when you have a lab/lecture in TEC 259
- Log on to D2L and record your preferred name and station number in *Assignments > Attendance*

Syllabus Review

Overview of the Python Language

Helpful References

- https://docs.python.org/3/tutorial/index.html

Introduction

- Python is a powerful interpreted language
- Can be run on any popular OS
- Used for:
  - Automating system administration tasks
  - Creating applications
  - Generating dynamic web pages

A First Program

```
#!/usr/bin/python3.11

name = input('What is your name?\n')
print('Hello, ' + name)
```

Note:
No type declaration
No main function
No semicolon

How to Run a Python Script (Method 1)

- To run a Python script directly from the command line:
  - Create a file, say *hello.py*, and add:
    #!/usr/bin/python3.11
    to **LINE 1**. Make sure there is **NO SPACE** before the #!
  - Make the file executable, e.g.,
    chmod 700 hello.py
  - Execute the file using
    ./hello.py
- *.py* extension is used by convention, but is neither necessary nor sufficient

How to Run a Python Script (Method 2)

- To run a Python script directly from the command line:
  - Create a file, say *hello.py*
  - Execute the file using
    python3.11 hello.py

Lab 1

- Set up the ICS 226 development environment:
  - SQP
  - Ubuntu App and VM
  - PyCharm IDE
  - GitHub

Some Common Data Types

- ***x = True***
- ***x = False***
- ***x = 1234***
- ***x = 'Hello World'***
- Unlike Java, variable types need not be declared
- The same variable can hold different types at different times

+=, -=, *=, and /=

- ***i***++ does not work in Python 3.11
- However, ***i*** += ***1*** does
- Note:
  - ***i*** += ***n*** is shorthand for ***i*** = ***i*** + ***n***
  - ***i*** -= ***n*** is shorthand for ***i*** = ***i*** - ***n***
  - ***i*** *= ***n*** is shorthand for ***i*** = ***i*** * ***n***
  - ***i*** /= ***n*** is shorthand for ***i*** = ***i*** / ***n***

** and /

- ***a*** ** ***b*** computes a^b
- ***a*** **/** ***b*** always returns a floating point number, even if *a* and *b* are whole numbers (unlike many other languages)

% and //

- ***a % b*** computes the remainder portion that results when dividing *a* by *b*
- e.g., ***16 % 8*** is 0 and ***16 % 9*** is 7
- ***a // b*** computes the integer portion that results when dividing *a* by *b*
- e.g., ***16 // 8*** is 2 and ***16 // 9*** is 1

_

- In interactive mode, the _ refers to the previous value, e.g.,

```
>>> 5 ** 3
125
>>> 2 + _
127
```

More on Strings

- Can use either apostrophes or quotation marks for strings, e.g.,
  x = 'Hello World' or ***x = "Hello World"***

- Concatenate strings using the + symbol, e.g.,
  'x' + 'y'

- Repeat strings using the * symbol, e.g.
  print('?' * 3)
  prints out
  ???

- Use *r* in front of apostrophes or quotation marks to turn off interpretation of the backslash, e.g.,
  print(r'Hello\nWorld')
  prints out
  Hello\nWorld

- Use ''' or """ to start and stop multi-line strings

More on Strings

- Access characters within a string using ***[]*** notation, e.g., if *x* is *Hello World* then
- ***x[1]*** is *e*
- ***x[-1]*** is *d*
- ***x[:5]*** is *Hello*
- ***x[6:]*** is *World*
- ***x[1:3]*** is *el*
- However, strings are immutable, so ***x[0] = 'h'*** will fail; we must instead create a new string

Exercises

- If ***x*** is *Hello World* then what is:
  - ***x[-2:]***
  - ***x[:-4]***
  - ***x[20]***
  - ***x[:20]***

How to Make Choices

- ***if*** _____:
  …
  elif _____:
  …
  else:
  …

- _____ represents a condition (e.g., ***x < 0***) and … represents the code that is to be executed if the condition is true

- ***elif*** is NOT a typo

- There can be multiple ***elif*** statements, but only one ***else***

- Both ***elif*** and ***else*** are optional

- Colons and consistent use of indentation is mandatory

if Example

```
#!/usr/bin/python3.11

n = int(input('Enter a number:\n'))  # Assume number entered
if n < 10:
    print('Value is less than 10')
elif n == 10:
    print('Value is equal to 10')
else:
    print('Value is greater than 10')
```

- ***int(s)*** converts string *s* to an integer; ***str(n)*** converts integer *n* to a string

- **#** precedes a comment

Exercises

- Write a program that prompts for the current speed and prints out either *Invalid* (assuming the input is less than 0), *OK* (assuming the input is in the range 0 to 50, inclusive), or *Too fast* (assuming the input is above 50). You can convert a string to a number using ***int(...)*** where ... is the string you wish to convert. Do not worry about invalid numbers for now.

- Write a program that prompts for a water temperature and then prints out one of *Below freezing point*, *Freezing point*, or *Above freezing point*, depending on the entered temperature. Assume 0 as the freezing point. Do not worry about invalid numbers for now.

Another Way to Make Choices

- ***match*** **_____:**
  case **_ _ _ _:**
  …

- Vaguely similar to a *switch* statement in Java:
  - _____ represents a variable
  - _ _ _ _ represents a pattern that the variable must match
  - … represents the code that is to be executed if the pattern is matched

match Example

```
#!/usr/bin/python3.11

p = input('Enter a new password: ')
match p:
    case '1234' | 'password':    # matches '1234' or 'password'
        print('Too common')

    case x if len(p) < 8:        # matches passwords of length < 8
        print(x, 'too short')

    case _:                      # matches everything
        print('OK')
```

Exercises

- Expand on the match example by rejecting passwords containing all digits. Hint: ***"1234".isnumeric()*** returns *True*.

How to Repeat While Condition Holds

- To repeat a statement multiple times, while a condition holds true:
- ***while*** **_____:**
  …
- _____ represents a condition (e.g., x < 0) and … represents the code that is to be executed if the condition is true.
- Colon and consistent use of indentation is mandatory

while Example

```
#!/usr/bin/python3.11

n = 1
while n < 10:
    n = n + 1
print(str(n))
```

- Prints out *10* when complete
- **Compare this program to the following program very carefully!**

Another ***while*** Example

```
#!/usr/bin/python3.11

n = 1
while n < 10:
    n = n + 1
    print(str(n))
```

- Prints out *2*, *3*, *4*, *5*, *6*, *7*, *8*, *9*, and then *10*
- **The only difference compared to the previous program is the indentation!**

Exercises

- Write a program that repeatedly prompts for a command until the *quit* command has been entered 3 times.

- Write a program that randomly generates a number in the range of 0 to 9 and then prompts for a number. The program should keep prompting until the user enters the randomly-generated number. To generate a random number, put ***import random*** at the top and use ***random.randrange(10)*** to generate a number in the range of 0 to 9. Do not worry about invalid numbers for now. However, note that ***input()*** returns a string whereas ***randrange()*** returns an int.

How to Repeat a Number of Times

- To repeat a statement a fixed number of times:
- ***for i in range(x, y):***

  …
- … represents the code that is to be executed repeatedly.
- In each iteration, i will hold a different value, going from x up to y - 1.
- Colon and consistent use of indentation is mandatory

for Examples

```
for i in range(10):
    print(str(i))
```

- Prints out *0*, *1*, *2*, *3*, *4*, *5*, *6*, *7*, *8*, *9*

```
for i in range(1, 3):
    print(str(i))
```

- Prints out *1* and then *2*

More *for* Examples

```
#!/usr/bin/python3.11

n = int(input('Enter a number:\n'))  # Assume valid number

s = 0
for i in range(1, n + 1):
    s += i
print('The numbers in 1 to', n, 'add up to', s)
```

More *for* Examples

```
#!/usr/bin/python3.11

n = int(input('Enter a number:\n'))  # Assume valid number

print('The numbers in1 to', n, 'add up to', sum(range(1, n + 1)))
```

Exercises

- Write a program that prints out the first 10 squares (1, 2, 4, 9, ...).
- Write a program that prints out 100 random numbers (each in the range of 0 to 99), then computes and prints out the average of them. Run this program multiple times to confirm that the average changes between most runs, and is close to 50.

How to Skip or Stop Iterations

- To stop executing a loop prematurely:
  - ***break***
- To stop the current iteration and go on to the next iteration of the loop:
  - ***continue***

break Example

```
#!/usr/bin/python3.11

num = int(input('Enter a number:\n'))  # Assume int > 1
i = 2
while i * i <= num:
    if num % i == 0:
        print(str(num) + ' is not a prime')
        break
    i = i + 1
```

- With input *12*, prints out *12 is not a prime*
  With input *13*, prints out *nothing*

else Example

```
#!/usr/bin/python3.11

num = int(input('Enter a number:\n'))  # Assume int > 1
i = 2
while i * i <= num:
    if num % i == 0:
        print(str(num) + ' is not a prime')
        break
    i = i + 1
else:
    print(str(num) + ' is a prime')
```

- With input *12*, prints out *12 is not a prime*
  With input *13*, prints out *13 is a prime*

- *else*, **when attached to a loop construct**, is called if no *break* occurs

continue Example

```
#!/usr/bin/python3.11


n = int(input('Enter a number:\n'))  # Assume valid number
m = n + 1
for i in range(1, m):
    if i % 2 == 0:
        continue

    print(str(i) + ' is odd')
```

- With input *12*, prints out
  1 is odd
  3 is odd
  5 is odd
  7 is odd
  9 is odd
  11 is odd

How to Repeat for Ea. Char in String

- To repeat a block of code a fixed number of times, once for each character in a string:
- ***for x in y:***
  …
- *y* must be a string.
- In each iteration, *x* will change to contain the next character in the string *y*.
- For example, if *y* is *abc*, *x* will first be *a*, then *b*, then *c*.

in Example

```
#!/usr/bin/python3.11

s = input('Enter a string:\n')
for ch in s:
    if (' ' <= ch <= '~') or (ch == '\n' or ch == '\t'):
        print(ch)
```

- Filters out unprintable characters
- Filtering useful when copying PDF or Windows files to Linux/ UNIX
- Relies on ASCII character order
- Can also use ***tr -cd '\11\12\40-\176' < infile > outfile*** to do this
- Note that ***tr*** uses octal, not decimal or hexadecimal value

in Example

Dec	Chr	Dec	Chr	Dec	Chr	Dec	Chr	Dec	Chr
0	NUL	26	SUB	52	4	78	N	104	h
1	SOH	27	ESC	53	5	79	O	105	i
2	STX	28	FS	54	6	80	P	106	j
3	ETX	29	GS	55	7	81	Q	107	k
4	EOT	30	RS	56	8	82	R	108	l
5	ENQ	31	US	57	9	83	S	109	m
6	ACK	32		58	:	84	T	110	n
7	BEL	33	!	59	;	85	U	111	o
8	BS	34	"	60	<	86	V	112	p
9	HT	35	#	61	=	87	W	113	q
10	LF	36	$	62	>	88	X	114	r
11	VT	37	%	63	?	89	Y	115	s
12	FF	38	&	64	@	90	Z	116	t
13	CR	39	'	65	A	91	[	117	u
14	SO	40	(	66	B	92	\	118	v
15	SI	41	)	67	C	93	]	119	w
16	DLE	42	*	68	D	94	^	120	x
17	DC1	43	+	69	E	95	_	121	y
18	DC2	44	,	70	F	96	`	122	z
19	DC3	45	-	71	G	97	a	123	{
20	DC4	46	.	72	H	98	b	124	|
21	NAK	47	/	73	I	99	c	125	}
22	SYN	48	0	74	J	100	d	126	~
23	ETB	49	1	75	K	101	e	127	DEL
24	CAN	50	2	76	L	102	f		
25	EM	51	3	77	M	103	g		

https://lookuptables.com

Exercises

- Write a program that prompts for a string and then prints out the number of digits in the string that are divisible by 5
- Write a program that prompts for a string and then prints out every 3rd character of that string