Content Covered

- Output Function
- Formatting
- Input
- Comment

Output

- When we need to display information to user, we use output function (**print**)
- Can print value, expression, variable
- Example:
  - print("printing is fun")
  - print(5)
  - print(5+2)
  - a = 4 * 5
  - print(a)

Output

- If we pass multiple argument to print, it displays them separating with space
- Example:
  - a = 5
  - b = 10
  - print("First Number is" + str(a))
  - print("First Number is", a)
  - print("Sum of", a, "and", b, "is", a + b)
  - print("Sum is", a + b, sep=':')

end

- By default, new line will be added after each print
- We can modify with end argument
- Example:
  - print(5)
  - print(10, end='\t')
  - print(15, end='---')
  - print()

Format

- For decorating output as desired
- Old style formatting supported but not used
- Syntax:
  - res = {[variable][:format_spec]}
- Example:
  - print("{} + {} = {}".format(5, 2, 5+2))
  - print("{1} + {0} = {2}".format(5, 2, 5+2))
  - a = 5
  - b = 4
  - c = a + b
  - print("{a}+{b}={c}".format(c=c, a=a, b=b))

Format Spec

- Place holder template is later filled with variable

Syntax:

- spec = {[fill][align][width][.precision][type]}
- Fill => Any Character (Default space)
- Align => <,>,^ (left, right, center)
- Width => Allocated space for print
- Type => b, d, o, x, X, e, E, f, %

Format Spec

Example:

- txt = "***“
- print("{txt:>5}".format(txt=txt))
- num = 20.2345
- print("{num:.2f}".format(num=num))
- val = 2
- print("{val:02}".format(val=val))
- print("{val:02}".format(val=str(val)))

F - String

- Similar to .format method
- All formatting spec applies
- We put f prior to string
- Example
  - num_1 = 1
  - num_2 = 3
  - res = num_1 / num_2
  - print(f"{num_1:02}/{num_2:02}={res:.2f}")

User Input

- For taking user input, we use **input**
- Value will be stored as str
- Type conversion is required for int & float
- Example:
  - usr_name = input(“Name: ”)
  - print(f”Hello {usr_name}”)

Comment

- Used to add detail on task
- Interpreter doesn't execute comment
- Can be done via # or three quotes

Gemini

- We can generate code in Collab using Gemini
- We can use Gemini to explain code and debug

Exercise