CS1301-Intro to Computing

Day 2

Welcome to CS1301

- Instructor:
  - Prof. Kearse (ikearse7@gatech.edu)
  - Office: CCB 255
  - Office Hours:
    - Tuesday @ 01:00 - 03:00 PM
- Head TA:
  - Chris Liding (zliding3@gatech.edu)

Review

- 1. Variables(Identifiers):
  - named memory locations that hold values
  - created based on syntax rules
- 2. Data types:
  - **str** (text in quotes), **int** (integer), **float** (decimal), **bool** (True, False)
- 3. Expressions:
  - a combination of values, variables, operators, and calls to functions.

Python basics

- **Function**: piece of prewritten code that performs an operation
- **Argument**: data given to a function
- **Comments**: notes of explanation within a program
  - Ignored by Python interpreter
  - Intended for a person reading the program's code
  - Begin with a # character
- **End-line comment**: appears at the end of a line of code
  - Typically explains the purpose of that line

Python basics

- **String literal**: a printable character written in a program
  - Must be enclosed in single (‘) or double (“) or triple (“’) quote marks
- **Numeric literal**: a numerical character written in a program
- **Variable**: name that represents a value stored in the computer memory
  - Must adhere to the naming rules
  - Variables can reference different values while program is running
  - A variable can refer to item of any type. A Variable that has been assigned to one type can be reassigned to another type
  - A variable can be passed as an argument to a function

Python basics

- **Formatted String Literal (f-String)**: a Python string formatting syntax
  - An f-string looks very much like a typical Python string except that it's prepended by the character f or F.
  - You can use single, double, or triple quotes to define an f-string
  - You can embed Python expressions directly inside them using curly braces {}
  - F-strings are faster than the older string formatting mechanisms, % formatting and str.format().
  - f-strings support extensive modifiers that can control the final appearance of the output string.

Python basics

- **Formatted String Literal (f-String)**: Syntax

  `f'{value:{width}.{precision}}'`

- where:
  - `value` is any expression that evaluates to a number
  - `width` specifies the number of characters used in total to display, but if value needs more space than the width specifies then the additional space is used.
  - `precision` indicates the number of characters used after the decimal point

```
>>> a = 10.1234
>>> f'{a:.2f}'
'10.12'

>> age = 20
>>> f'{age=}'
age = 20
```

```
x = 1000000
>>> print(f'{x:,}')
1,000,000

>>> x = 20.123
>>> print(f'{x:.1f}%')
20.1%
```

Python function: **`type`**

- If you are not sure what class a value falls into, Python has a function called **type** which can tell you.

```
>>> type("Hello, World!")
<class ‘str’>
>>> type('This is a string.')
<class 'str'>
>>> type("And so is this.")
<class 'str'>
>>> type("""and this.""")
<class 'str'>
>>> type('''and even this...''')
<class 'str'>
```

```
>>> type("17")
<class 'str'>
>>> type("3.2")
<class 'str’>
>>> type(17)
<class 'int'>
>>> type(3.2)
<class 'float'>
```

Python function: **`input`**

- Reads input from the Keyboard
- Always returns the data as a string
- Format: `variable = input(prompt)`
  - prompt is typically a string instructing the user to enter a value
  - Does not automatically display a space after the prompt

```
>>>name = input("Enter your name: ")
Enter your name: Renee
>>>type(name)
<class 'str’>

>>>name = input("Enter your age: ")
Enter your age: 19
>>>type(age)
<class 'str'>
```

Python function: **`print`**

- Displays a line of output
- The newline character is automatically inserted at the end of a line
- Format: `print(value)`
  - value can be an expression, string literal, numeric literal, formatted string literal
  - Multiple values are separated my commas

```
>>> print('gpa’)
gpa

>>> quantity, item, price = 6, 'bananas', 1.74
>>> print(f'Price per item is ${price/quantity}')
Price per item is $0.29
```

```
>>> print(gpa)
4.0
>>> print(4+26)
30
>>> print('year is',2023)
year is 2023
```

Type converter functions: str(), float(), int(), bool()

- Variables and values can be **type cast** to a different data type
- Use a data type function to change the type of an expression

```
>>> int(3.14)
3
>>> int(-3.999)
-3
>>> int(minutes / 60)
10
>>> int("2345")
2345
>>> str(17)
'17’
>>> str(123.45)
'123.45'
```

```
>>> float(17)
17.0
>>> float("123.45")
123.45

>>> int("23 bottles") #str is not numeric characters
Traceback (most recent call last):
File "<interactive input>", line 1, in
<module>
ValueError: invalid literal for int()
with base 10: '23 bottles’
```

Evaluating expressions

- An expression is a combination of values, variables, operators, and calls to functions.
  - If you type an expression at the Python prompt, the interpreter evaluates it and displays the result.
- The evaluation of an expression produces a value, which is why expressions can appear on the right-hand side of assignment statements.
  - A value all by itself is a simple expression, and so is a variable.
- When more than one operator appears in an expression, the order of evaluation depends on the rules of precedence.
  - Python follows the same precedence rules for its mathematical operators that mathematics does.

Order of precedence

1. evaluate the arithmetic operators (PEMDAS).

Precedence Level	Operator	Explanation
1 (highest)	( )	Parentheses
2	**	Exponentiation
3	-a, +a	Negative, positive argument
4	* , / , // , %, @	Multiplication, division, floor division, modulus, at
5	+, -	Addition, subtraction

Order of precedence

2. evaluate the relational operators.

Precedence Level	Operator	Explanation
6	< , <=, >, >=, ==, !=	Less than, less than or equal, greater, greater or equal, equal, not equal

3. evaluate the logical operators.

Precedence Level	Operator	Explanation
7	not	Boolean Not
8	and	Boolean And
9	or	Boolean Or

Order of precedence: Division

- Division - **/** result is always a floating-point number.

```
>>> 7 / 4
1.75
```

- Floor Division - **//** result is always a whole number when the first number is divided by the second.

```
>>> 7 // 4
1
```

Order of precedence: Modulus

- Modulus - **%** works on integers and gives the remainder when the first number is divided by the second.
  - Ex: 7 divided by 3 is 2 with a remainder of 1.

```
>>> 7 // 3
2
>>> 7 % 3
1
```

Designing a Program

- Programs must be designed before they are written
- Program development cycle:
  - Design the program
  - Write the code
  - Correct syntax errors
  - Test the program
  - Correct logic errors

Designing a Program (cont'd.)

- Design is the most important part of the program development cycle
- Understand the task that the program is to perform, or the problem to be solved
  - Ask questions about program details to get a sense what the program is supposed to do
- Determine the steps that must be taken to perform the task
  - Break down required task into a series of steps
  - Create an algorithm, listing logical steps that must be taken
- Algorithm: set of well-defined logical steps that must be taken to perform a task

The IDLE Programming Environment

- IDLE (Integrated Development Program): single program that provides tools to write, execute and test a program
  - Automatically installed when Python language is installed
  - Runs in interactive mode
  - Has built-in text editor with features designed to help write Python programs
- IDLE has a debugger that you launch from the Shell.
  - IDLE Debugger Tutorial - Python IDE (by Eleros Vecchio)
  - Debugging in IDLE (by Uconn UPE)
- IDLE can display line numbers in script mode.
  - Use the options menu

After Class

1. Read textbook Chapter 1 and Chapter 2 and Chapter 4
2. Review lecture notes and code from today
3. Practice Python

Time to program

1. Launch Canvas
   - Select this course KEARSE- CS1301-A/C
   - Click Files - Lecture notes - **02notes.py, 02code.py**
   - **download the files to your CS1301 folder**
2. Launch IDLE
   - Click on the File - Open
   - Navigate to your CS1301 folder
   - Click on the downloaded file for today
   - Click the Open button