od8/29 CSP P4E chp2 notes:

Overview

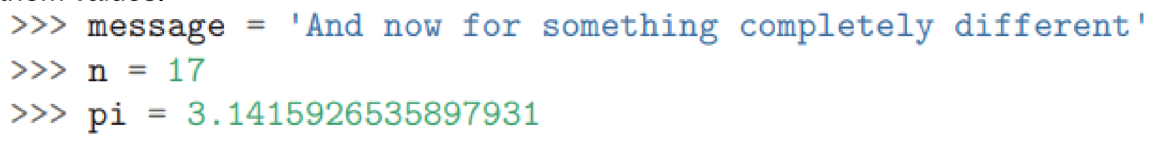
Chapter 2: Variables, Expressions, and Statements

2.1 Values and Types

- **Definition of Values:** A value is one of the basic things a program works with, like a letter or a number.
- **Types of Values:**
  - *Integers* are just numbers. Example: 2
  - *Strings,* so called because it contains a "string" of letters.You can identify strings because they are enclosed in quotation marks. Example: "Hello, World!"
  - *Floating points,* are numbers with a decimal point, they belong to a type called `float`
- Even if you put "17" or "3.2" with quotation marks, they become strings.
- Commas are not a legal integer in Python, it won't give an error but for example: print(1,000,000) gives you "1 0 0".

2.2 Variables

- **Definition of Variable:** A variable is a name that refers to a value.
- **Assignment Statement:** An assignment statement creates new variables and gives them values.

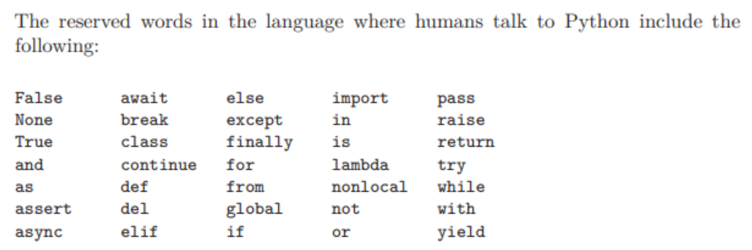


```
>>> message = 'And now for something completely different'
>>> n = 17
>>> pi = 3.1415926535897931
```

-

2.3 Variable Names and Keywords

- **Naming Rules:**They can contain both letters and numbers, but can't start with a number. It is legal to use uppercase letters, but isn't' the best idea.
- **More rules**: Underscores are common, but not recommended to start a word with. Cannot contain illegal characters like "@". Cannot include reserved 35 words.



The reserved words in the language where humans talk to Python include the following:

False	await	else	import	pass
None	break	except	in	raise
True	class	finally	is	return
and	continue	for	lambda	try
as	def	from	nonlocal	while
assert	del	global	not	with
async	elif	if	or	yield

-
- **Python Keywords:**^

2.4 Statements

- **Definition:** A *statement* is a unit of code that the Python interpreter can execute.
- A *script* usually contains a sequence of statements. If there is more than one statement, the results appear one at a time as the statements execute.

2.5 Operators and Operands

- **Operators:** Operators are special symbols that represent computations like addition and multiplication +, -, *, /, and ** perform addition, subtraction, multiplication, division, and exponentiation,
- **Operands:** One of the values on which an operator operates.
- In Python 2, when dividing 2 integers it would remove the float to keep it whole, so to emulate it now, To obtain the same answer in Python 3 use floored (// integer) division.

2.6 Expressions

- **Definition:** An *expression* is a combination of values, variables, and operators. A value all by itself is considered an expression, and so is a variable, so the following are all legal expressions
- **Examples:**
- in a script, an expression all by itself doesn't do anything! This is a common source of confusion for beginners

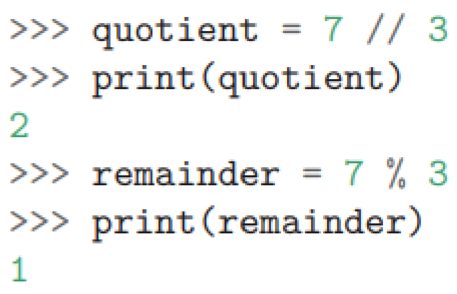
2.7 Order of Operations

- **PEMDAS:** Parentheses, Exponents, Multiplication and Division, Addition and Subtraction

2.8 Modulus Operator

- **Definition:** The modulus operator works on integers and yields the remainder when the first operand is divided by the second. The modulus operator is a percent sign (%).

- **Examples:**



```
>>> quotient = 7 // 3
>>> print(quotient)
2
>>> remainder = 7 % 3
>>> print(remainder)
1
```

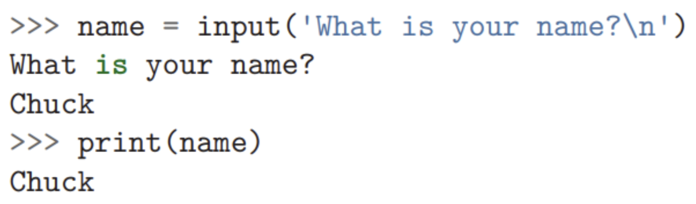
- **Other uses:** For example, you can check whether one number is divisible by another: if x % y is zero, then x is divisible by y
- **More:** You can also extract the right-most digit or digits from a number. For example, x % 10 yields the right-most digit of x (in base 10). Similarly, x % 100 yields the last two digits.

2.9 String Operations

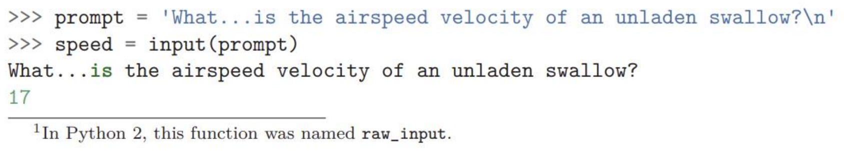
- **Concatenation:** + performs concatenation, Joining the strings by linking them end to end. . If you make a variable with 'integer' and 2 of them. It would add like this " '69' +'1' =691
- **Repetition:** The * operator also works with strings by multiplying the content of a string by an integer.   first = 'Test ', second = 3, print(first * second), Test Test Test
- **Indexing:** [How to access specific characters in a string]

2.10 Asking the User for Input

- **Input Function:**

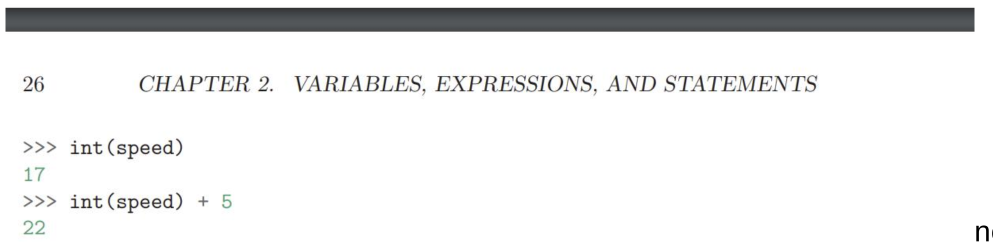


```
>>> name = input('What is your name?\n')
What is your name?
Chuck
>>> print(name)
Chuck
```



```
>>> prompt = 'What...is the airspeed velocity of an unladen swallow?\n'
>>> speed = input(prompt)
What...is the airspeed velocity of an unladen swallow?
17
```

1In Python 2, this function was named raw_input.



26 *CHAPTER 2. VARIABLES, EXPRESSIONS, AND STATEMENTS*

```
>>> int(speed)
17
>>> int(speed) + 5
22
```

- newline=n

-
- However if the user types something other than a string of digits, you get an error

2.11 Comments

- **Single-Line Comments:** These notes are called comments, and in Python they start with the # symbol: # compute the percentage of the hour that has elapsed
- **Uses :**Comments are most useful when they document non-obvious features of the code.

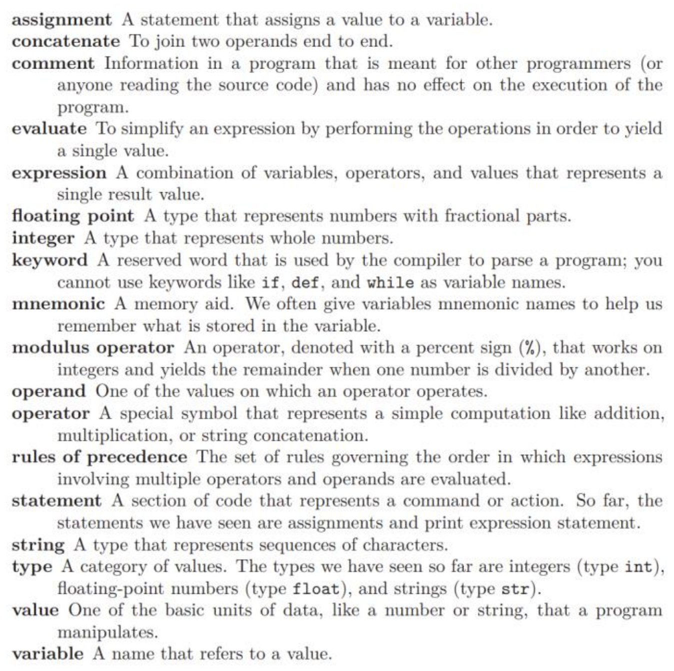
2.12 Choosing Mnemonic Variable Names

- **Wisely chosen variable names "mnemonic variable names".**]
- **Examples:** But if our program is truly about reading data and looking for words in the data, pizza and slice are very un-mnemonic variable names. Choosing them as variable names distracts from the meaning of the program.

2.13 Debugging

- **Common Errors:**
- cant have spaces or illegal characters in variables
- Variable names are case sensitive, so LaTeX is not the same as latex.
- At this point, the most likely cause of a semantic error is the order of operations.
- **Debugging Techniques:** Techniques for finding and fixing errors in code

2.14 Glossary



assignment A statement that assigns a value to a variable.

concatenate To join two operands end to end.

comment Information in a program that is meant for other programmers (or anyone reading the source code) and has no effect on the execution of the program.

evaluate To simplify an expression by performing the operations in order to yield a single value.

expression A combination of variables, operators, and values that represents a single result value.

floating point A type that represents numbers with fractional parts.

integer A type that represents whole numbers.

keyword A reserved word that is used by the compiler to parse a program; you cannot use keywords like `if`, `def`, and `while` as variable names.

mnemonic A memory aid. We often give variables mnemonic names to help us remember what is stored in the variable.

modulus operator An operator, denoted with a percent sign (`%`), that works on integers and yields the remainder when one number is divided by another.

operand One of the values on which an operator operates.

operator A special symbol that represents a simple computation like addition, multiplication, or string concatenation.

rules of precedence The set of rules governing the order in which expressions involving multiple operators and operands are evaluated.

statement A section of code that represents a command or action. So far, the statements we have seen are assignments and print expression statement.

string A type that represents sequences of characters.

type A category of values. The types we have seen so far are integers (type `int`), floating-point numbers (type `float`), and strings (type `str`).

value One of the basic units of data, like a number or string, that a program manipulates.

variable A name that refers to a value.

- **Key Terms:**

2.15 Exercises