Chapter 4 Selections

Motivations

If you asked the user for the radius of a circle, and they entered a negative value ……

Then you computed: area = pi * radius ** 2

You would get the same answer as when they entered the positive value of the same number.

Yet ……. a negative radius is invalid.
How can you deal with this situation?

Objectives

- To write Boolean expressions by using comparison operators
- To implement selection control by using one-way if statements
- To implement selection control by using two-way **if .. else** statements
- To implement selection control with nested **if ... elif ... else** statements
- To avoid common errors in **if** statements
- To combine conditions by using logical operators (**and**, **or**, and **not**)
- To understand the rules governing operator precedence and associativity
- To generate random numbers by using the **random.randint(a, b)** or **random.random()** functions

Boolean Data Types

Often in a program you need to compare two values, such as whether *i* is greater than *j*.

There are six comparison operators (also known as relational operators) that can be used to compare two values. The result of the comparison is a Boolean value: true or false.

Comparison (Relational) Operators

Operator	*Name*
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	equal to
!=	not equal to

Problem: A Simple Math Learning Tool

This example creates a program to let a first grader practice additions. The program randomly generates two single-digit integers num1 and num2 and displays a question such as “What is 7 + 9?” to the student. After the student types the answer, the program displays a message to indicate whether the answer is true or false.

AdditionQuiz

Note: Indentation

```
if i > 0:
print("i is positive")
```

(a) Wrong

```
if i > 0:
    print("i is positive")
```

(b) Correct

Simple if Demo

Write a program that prompts the user to enter an integer.

If the number is a multiple of 5, print HiFive.

If the number is divisible by 2, print HiEven.

SimpleIfDemo

The Two-way if Statement

```
if boolean-expression:
    statement(s)-for-the-true-case
else:
    statement(s)-for-the-false-case
```

if...else Example

```
radius = eval(input("Enter radius: "))
if radius >= 0:
    area = radius * radius * math.pi
    print("area for circle of radius", radius, "is", area)
else:
    print("Negative input")
```

Problem: An Improved Math Learning Tool

This example creates a program to teach a first grade child how to learn subtraction.

The program randomly generates two single-digit integers, num1 and num2, with num1 > num2

It displays a question such as

“What is 9 – 2?” to the student.

After the student types the answer in the input dialog box, the program displays a message dialog box to indicate whether the answer is correct.

SubtractionQuiz

Multiple Alternative if Statements

```
if score >= 90.0:
    grade = 'A'
else:
    if score >= 80.0:
        grade = 'B'
  else:
      if score >= 70.0:
          grade = 'C'
      else:
          if score >= 60.0:
              grade = 'D'
          else:
              grade = 'F'
```

(a)

Equivalent

This is better

```
if score >= 90.0:
    grade = 'A'
elif score >= 80.0:
    grade = 'B'
elif score >= 70.0:
    grade = 'C'
elif score >= 60.0:
    grade = 'D'
else:
    grade = 'F'
```

(b)

Trace if-else statement

Suppose score is 70.0

The condition is false

```
if score >= 90.0:
    grade = 'A'
elif score >= 80.0:
    grade = 'B'
elif score >= 70.0:
    grade = 'C'
elif score >= 60.0:
    grade = 'D'
else:
    grade = 'F'
```

Trace if-else statement

Suppose score is 70.0

The condition is false

```
if score >= 90.0:
    grade = 'A'
elif score >= 80.0:
    grade = 'B'
elif score >= 70.0:
    grade = 'C'
elif score >= 60.0:
    grade = 'D'
else:
    grade = 'F'
```

Trace if-else statement

Suppose score is 70.0

The condition is true

```
if score >= 90.0:
    grade = 'A'
elif score >= 80.0:
    grade = 'B'
elif score >= 70.0:
    grade = 'C'
elif score >= 60.0:
    grade = 'D'
else:
    grade = 'F'
```

Trace if-else statement

Suppose score is 70.0

grade is C

```
if score >= 90.0:
    grade = 'A'
elif score >= 80.0:
    grade = 'B'
elif score >= 70.0:
    grade = 'C'
elif score >= 60.0:
    grade = 'D'
else:
    grade = 'F'
```

Trace if-else statement

Suppose score is 70.0

Exit the if statement

```
if score >= 90.0:
    grade = 'A'
elif score >= 80.0:
    grade = 'B'
elif score >= 70.
    grade = 'C'
elif score >= 60.0:
    grade = 'D'
else:
    grade = 'F'
```

Logical Operators

Operator	Description
not	logical negation
and	logical conjunction
or	logical disjunction

Truth Table for Operator and

p1	p2	p1 and p2	Example (assume age = 24, gender = 'F')
False	False	False	(age > 18) and (gender == 'F') is True, because (age > 18) and (gender == 'F') are both True.
False	True	False	
True	False	False	(age > 18) and (gender != 'F') is False, because (gender != 'F') is False.
True	True	True	

Truth Table for Operator or

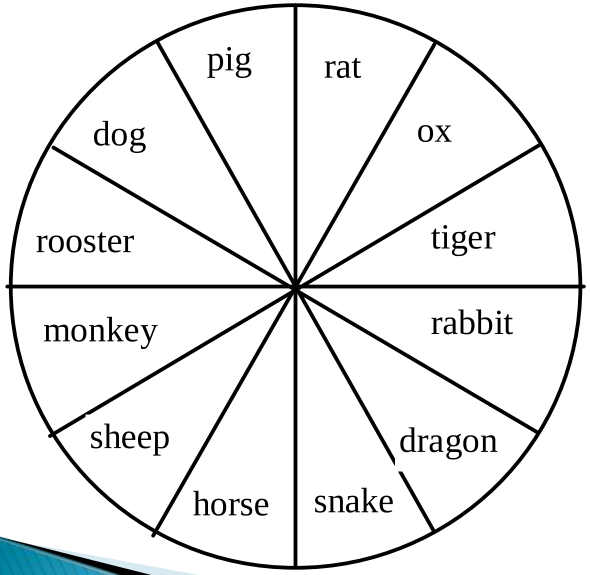
p1	p2	p1 or p2	Example (assume age = 24, gender = 'F')
False	False	False	(age > 34) or (gender == 'F') is true, because (gender == 'F') is True.
False	True	True	
True	False	True	(age > 34) or (gender == 'M') is False, because (age > 34) and (gender == 'M') are both Talse.
True	True	True	

Truth Table for Operator not

p	not p	Example (assume age = 24, gender = 'F')
True	False	not (age > 18) is False, because (age > 18) is True.
False	True	not (gender == 'M') is True, because (grade == 'M') is False.

Example

Now let us write a program to find out the Chinese Zodiac sign for a given year. The Chinese Zodiac sign is based on a 12-year cycle, each year being represented by an animal: rat, ox, tiger, rabbit, dragon, snake, horse, sheep, monkey, rooster, dog, and pig, in this cycle.



year % 12 =

0: monkey
1: rooster
2: dog
3: pig
4: rat
5: ox
6: tiger
7: rabbit
8: dragon
9: snake
10: horse
11: sheep

ChineseZodiac

Problem: Body Mass Index

Body Mass Index (BMI) is a measure of health on weight. It can be calculated by taking your weight in kilograms and dividing by the square of your height in meters. The interpretation of BMI for people 16 years or older is as follows:

BMI	Interpretation
Below 18.5	Underweight
18.5-24.9	Normal
25.0-29.9	Overweight
Above 30.0	Obese

ComputeBMI

Examples

Here is a program that checks whether a number is:

- divisible by 2 and 3
- divisible by 2 or 3
- divisible by 2 or 3 but not both

TestBooleanOperators

Problem: Determining Leap Year?

This program first prompts the user to enter a year as an int value and checks if it is a leap year.

A year is a leap year if it is divisible by 4 but not by 100, or it is divisible by 400.

```
(year % 4 == 0 and year % 100 != 0) or

(year % 400 == 0)
```

LeapYear

Problem: Lottery

Write a program that randomly generates a lottery of a two-digit number, prompts the user to enter a two-digit number, and determines whether the user wins according to the following rule:

- If the user input matches the lottery in exact order, the award is $10,000.
- If the user input matches the lottery, the award is $3,000.
- If one digit in the user input matches a digit in the lottery, the award is $1,000.

Lottery

Operator Precedence

- +, -
- **
- not
- *, /, //, %
- +, -
- <, <=, >, >=
- ==, !=
- and
- or
- =, +=, -=, *=, /=, //=, %= (Assignment operator)

Operator Precedence and Associativity

The expression in the parentheses is evaluated first. (Parentheses can be nested, in which case the expression in the inner parentheses is executed first.) When evaluating an expression without parentheses, the operators are applied according to the precedence rule and the associativity rule.

If operators with the same precedence are next to each other, their associativity determines the order of evaluation. All binary operators except assignment operators are left-associative.

Operator Associativity

When two operators with the same precedence are evaluated, the *associativity* of the operators determines the order of evaluation. All binary operators except assignment operators are *left–associative*.

a – b + c – d is equivalent to

((a – b) + c) – d

Assignment operators are *right–associative*. Therefore, the expression

a = b += c = 5 is equivalent to

a = (b += (c = 5))