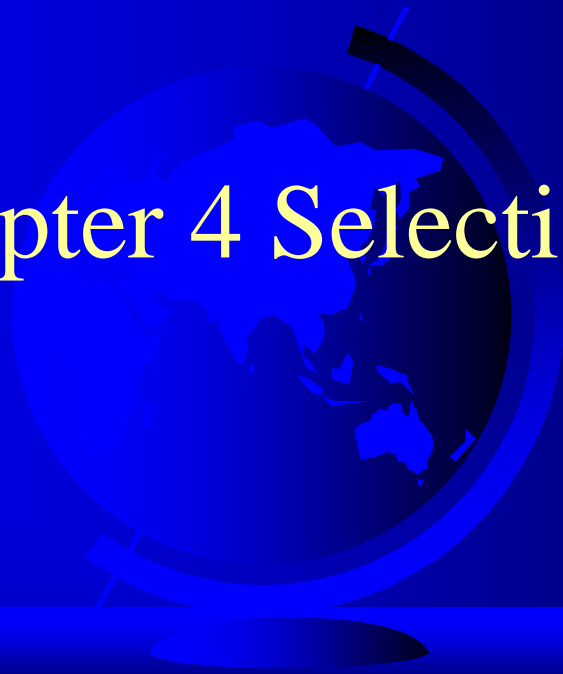


# Chapter 4 Selections



# Motivations

If you assigned a **negative value for radius** in Listing 2.1, `ComputeArea.py`, the program would **print an invalid result**. If the radius is **negative**, you **don't want the program to compute the area**. How can you deal with this situation?



# Objectives



- ❑ To write Boolean expressions by using comparison operators (§4.2).
- ❑ To generate random numbers by using the **random.randint(a, b)** or **random.random()** functions (§4.3).
- ❑ To program with Boolean expressions (**AdditionQuiz**) (§4.3).
- ❑ To implement selection control by using one-way **if** statements (§4.4)
- ❑ To program with one-way **if** statements (**GuessBirthday**) (§4.5).
- ❑ To implement selection control by using two-way **if .. else** statements (§4.6).
- ❑ To implement selection control with nested **if ... elif ... else** statements (§4.7).
- ❑ To avoid common errors in **if** statements (§4.8).
- ❑ To program with selection statements (§4.9–4.10).
- ❑ To combine conditions by using logical operators (**and**, **or**, and **not**) (§4.11).
- ❑ To use selection statements with combined conditions (**LeapYear**, **Lottery**) (§§4.12–4.13).
- ❑ To write expressions that use the conditional expressions (§4.14).
- ❑ To understand the rules governing operator precedence and associativity (§4.15).

# 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**.

```
b = (1 > 2)
```



# Comparison Operators

| <i>Operator</i> | <i>Name</i>              |
|-----------------|--------------------------|
| <               | less than                |
| <=              | less than or equal to    |
| >               | greater than             |
| >=              | greater than or equal to |
| ==              | equal to                 |
| !=              | not equal to             |



# Comparison Operators

```
x = 5  
y = 8
```

```
print("x == y:", x == y)  
print("x != y:", x != y)  
print("x < y:", x < y)  
print("x > y:", x > y)  
print("x <= y:", x <= y)  
print("x >= y:", x >= y)
```

## Output

```
x == y: False  
x != y: True  
x < y: True  
x > y: False  
x <= y: True  
x >= y: False
```



# 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 number1 and number2 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

Run

# if Statements

- Python has several types of selection statements:
  - one-way **if** statements,
  - two-way **if-else** statements,
  - nested **if** statements,
  - multi-way **if-elif-else** statements and
  - conditional expressions





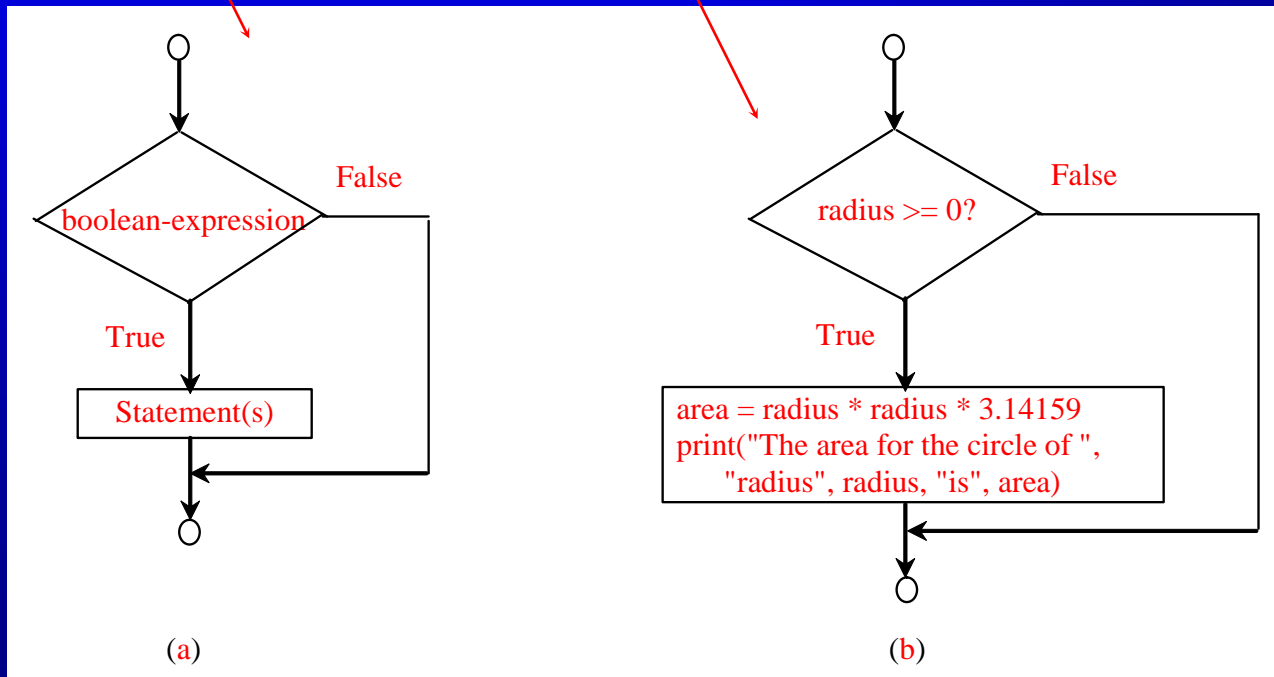
# One-way if Statements

if boolean-expression:  
statement(s)

if radius  $\geq$  0:

```
area = radius * radius * 3.14159
```

```
print("The area for the circle of radius",  
      radius, "is", area)
```



*A one-way if statement executes the statements if the condition is true.*

# Note

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

(a) Wrong

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

(b) Correct

- The statement(s) must be indented **at least one space** to the right of the if keyword and each statement must be indented using **the same number of spaces**.
- For consistency, we indent it four spaces in this book.



# 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.



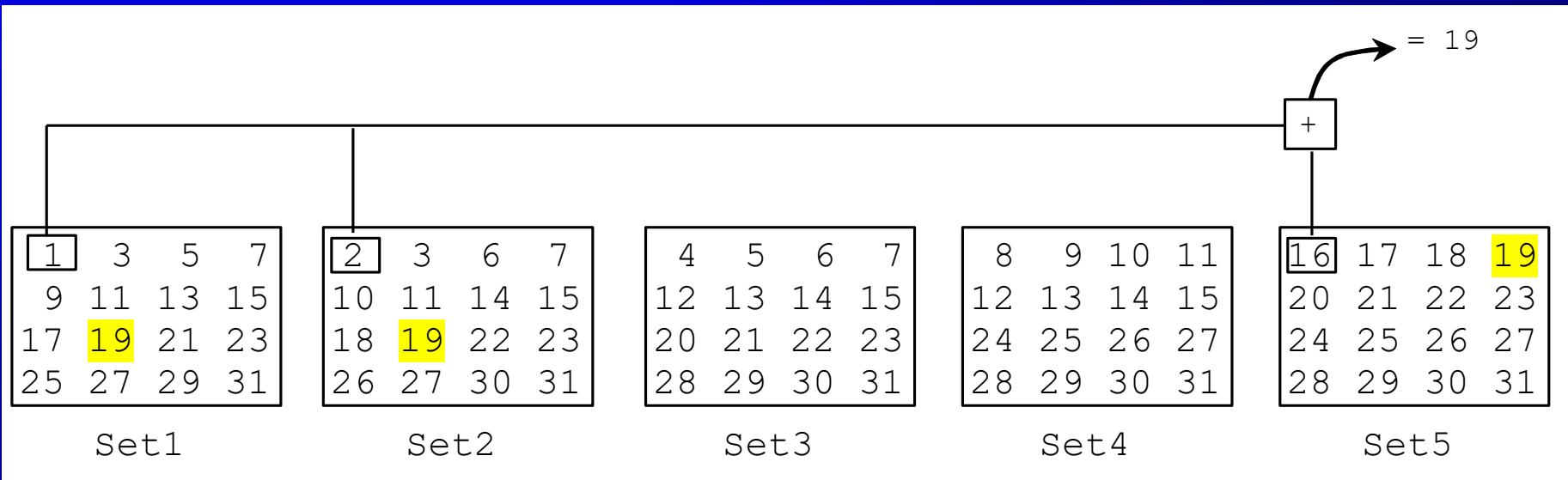
SimpleIfDemo



Run

# Problem: Guessing Birthday

- ❑ You can find out the date of the month when your friend was born by **asking five questions**.
- ❑ Each question asks whether the day is **in one of the five sets of numbers**.
- ❑ The birthday is the **sum of the first numbers** in the sets where the date appears.
- ❑ The program can guess your birth date. Run to see how it works.



GuessBirthday

Run

# Mathematics Basis for the Game

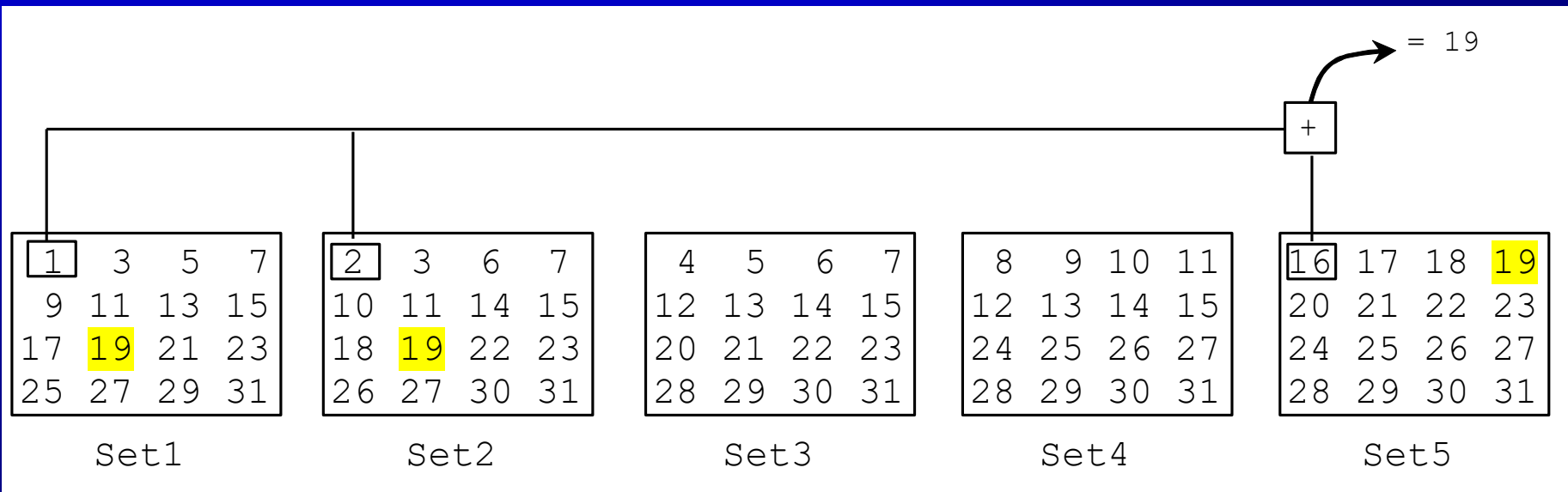
19 is 10011 in binary. 7 is 111 in binary. 23 is 11101 in binary

|   |  |  |
|---|--|--|
| $\begin{array}{r} 10000 \\ 10 \\ + 1 \\ \hline 10011 \\ 19 \end{array}$ | $\begin{array}{r} 00110 \\ 10 \\ + 1 \\ \hline 00111 \\ 7 \end{array}$ | $\begin{array}{r} 10000 \\ 1000 \\ 100 \\ + 1 \\ \hline 11101 \\ 23 \end{array}$ |
|---|--|--|

| Decimal | Binary |
|---------|--------|
| 1       | 00001  |
| 2       | 00010  |
| 3       | 00011  |
| ...     |        |
| 19      | 10011  |
| ...     |        |
| 31      | 11111  |

|                       |       |       |
|-----------------------|-------|-------|
| $b_5 0000$            |       | 10000 |
| $b_4 000$             |       | 1000  |
| $b_3 00$              | 10000 | 100   |
| $b_2 0$               | 10    | 10    |
| $+ b_1$               | $+ 1$ | $+ 1$ |
| $b_5 b_4 b_3 b_2 b_1$ | 10011 | 11111 |
|                       | 19    | 31    |



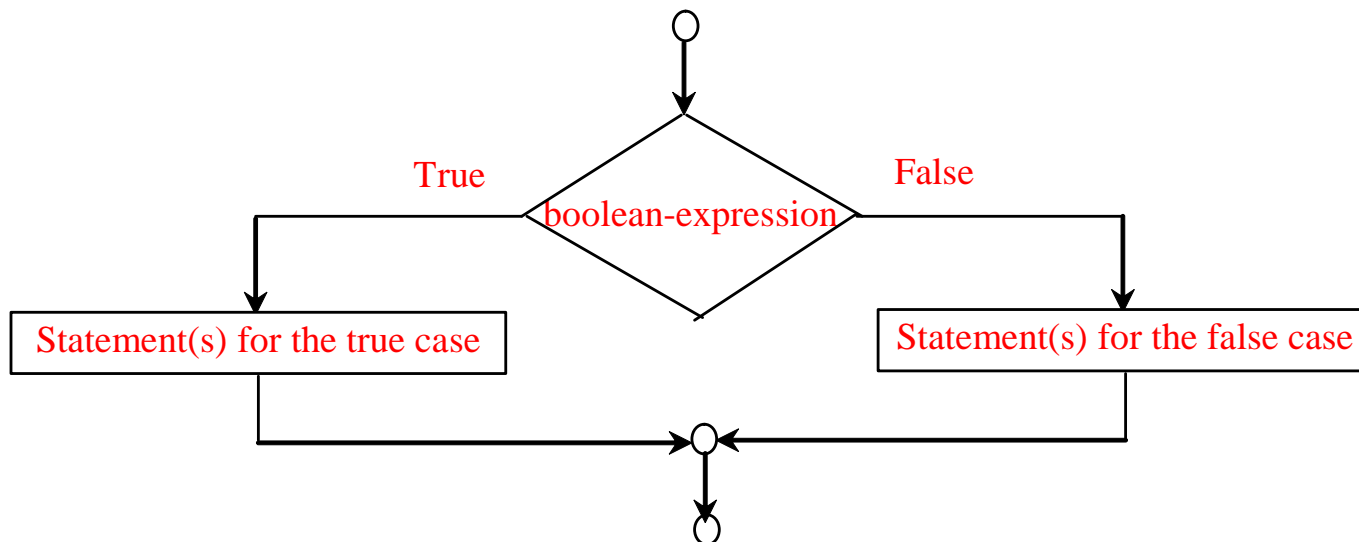
# The Two-way `if` Statement

**if** boolean-expression:

statement (s) -for-the-true-case

**else:**

statement (s) -for-the-false-case



# if...else Example

**if** radius  $\geq$  0:

```
    area = radius * radius * math.pi
```

```
    print("The area for the 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 subtractions**. The program **randomly generates two single-digit integers** number1 and number2 with  $\text{number1} > \text{number2}$  and **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

Run



# Nested if

- *One **if** statement can be placed inside another **if** statement to form a nested **if** statement.*

```
if i > k:  
    if j > k:  
        print("i and j are greater than k")  
else:  
    print("i is less than or equal to k")
```



# Multiple Alternative (MultiWay) 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)

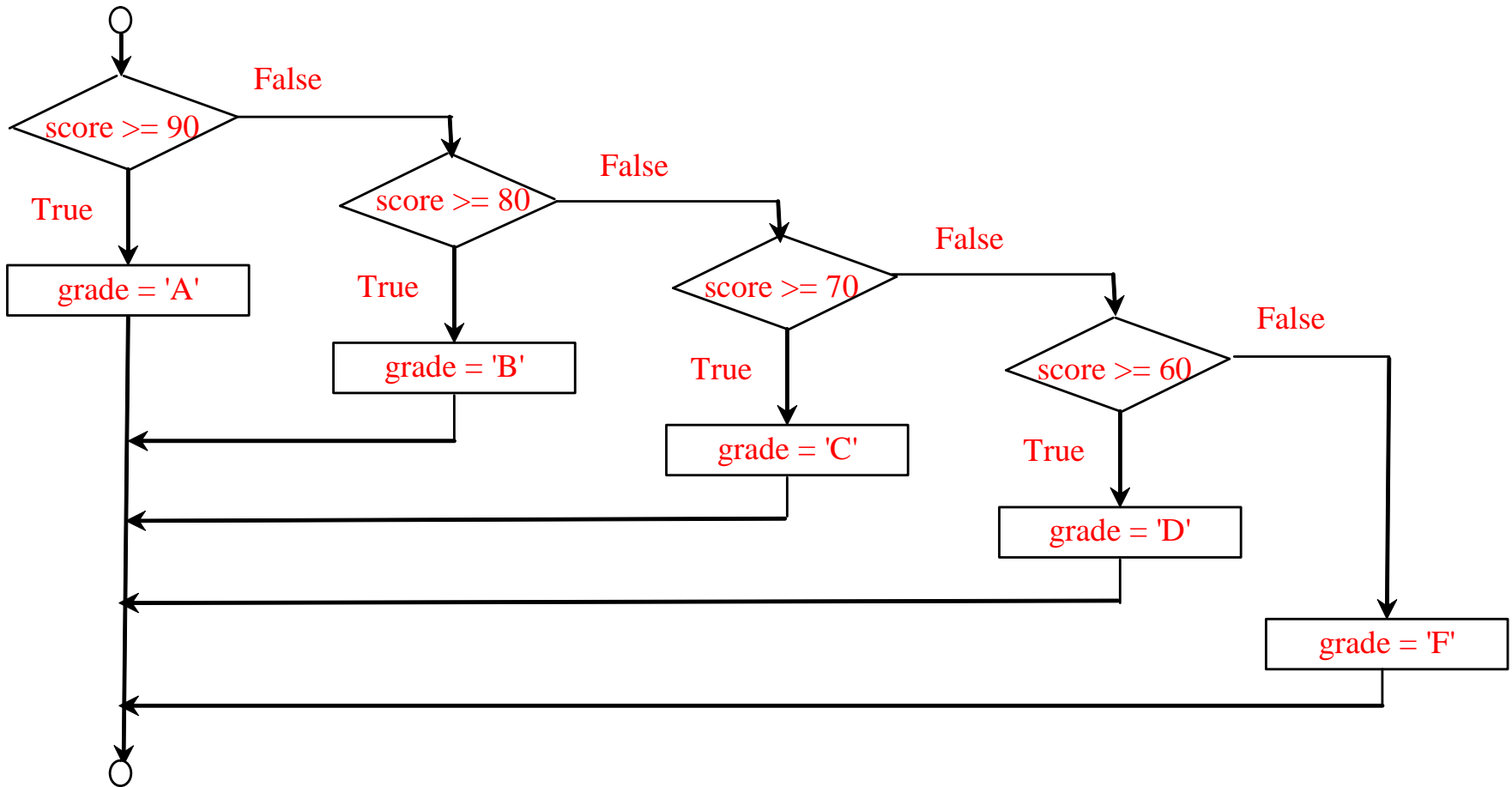
Equivalent

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)

# Flowchart



# 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

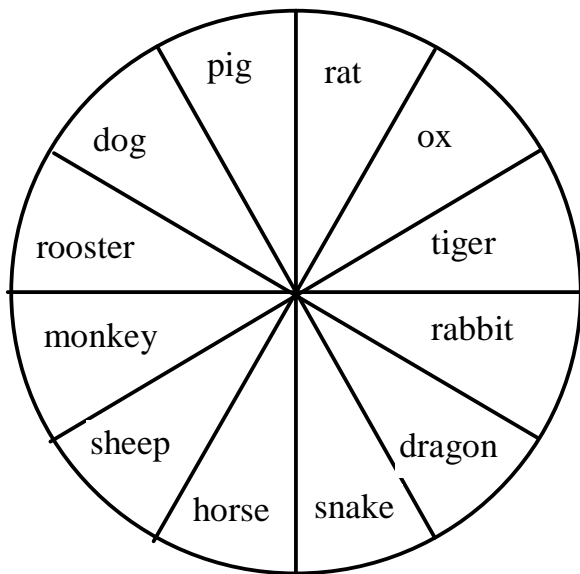
```
if score >= 90.0:  
    grade = 'A'  
elif score >= 80.0:  
    grade = 'B'  
elif score >= 70.0:  
    grade = 'C'  
elif score >= 60.0:  
    grade = 'D'  
else:  
    grade = 'E'
```





# 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

Run

# Common Errors

Most common errors in selection statements are caused by incorrect indentation. Consider the following code in (a) and (b).

```
radius = -20

if radius >= 0:
    area = radius * radius * 3.14
print("The area is", area)
```

(a) Wrong

```
radius = -20

if radius >= 0:
    area = radius * radius * 3.14
    print("The area is", area)
```

(b) Correct



# Nested If

Which if clause is matched by the else clause? The indentation indicates that the else clause matches the first if clause in (a) and the second if clause in (b).

```
i = 1
j = 2
k = 3

if i > j:
    if i > k:
        print('A')
else:
    print('B')
```

(a)

```
i = 1
j = 2
k = 3

if i > j:
    if i > k:
        print('A')
    else:
        print('B')
```

(b)

TIP: The code can be simplified by assigning the test value directly to the variable, as shown in (b)

```
if number % 2 == 0:
    even = True
else:
    even = False
```

(a)

Equivalent

This is shorter

```
even = number % 2 == 0
```

(b)

# 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:

Formula:  $\text{weight (kg)} / [\text{height (m)}]^2$

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

[ComputeBMI](#)

Run

# Problem: Computing Taxes

The US federal **personal income tax** is calculated based on the **filing status** and taxable income. There are four **filing statuses**: single filers, married filing jointly, married filing separately, and head of household. The tax rates for 2009 are shown below.

If you are, say, single with a taxable income of \$10,000, the first \$8,350 is taxed at **10%** and the other \$1,650 is taxed at **15%**. So, your tax is \$1,082.50.

| Marginal Tax Rate | Single                | Married Filing Jointly or Qualified Widow(er) | Married Filing Separately | Head of Household     |
|-------------------|-----------------------|---|---------------------------|-----------------------|
| 10%               | \$0 – \$8,350         | \$0 – \$16,700                                | \$0 – \$8,350             | \$0 – \$11,950        |
| 15%               | \$8,351 – \$33,950    | \$16,701 – \$67,900                           | \$8,351 – \$33,950        | \$11,951 – \$45,500   |
| 25%               | \$33,951 – \$82,250   | \$67,901 – \$137,050                          | \$33,951 – \$68,525       | \$45,501 – \$117,450  |
| 28%               | \$82,251 – \$171,550  | \$137,051 – \$208,850                         | \$68,525 – \$104,425      | \$117,451 – \$190,200 |
| 33%               | \$171,551 – \$372,950 | \$208,851 – \$372,950                         | \$104,426 – \$186,475     | \$190,201 – \$372,950 |
| 35%               | \$372,951+            | \$372,951+                                    | \$186,476+                | \$372,951+            |

# Problem: Computing Taxes, cont.

```
if status == 0:  
    # Compute tax for single filers  
elif status == 1:  
    # Compute tax for married filing jointly  
elif status == 2:  
    # Compute tax for married filing separately  
elif status == 3:  
    # Compute tax for head of household  
else:  
    # Display wrong status
```

ComputeTax

Run

# Logical Operators

| <b>Operator</b> | <b>Description</b>  |
|-----------------|---------------------|
| not             | logical negation    |
| and             | logical conjunction |
| or              | logical disjunction |



# 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 (gender == 'M') is False. |





# 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     | (age > 18) and (gender == 'F') is True, because (age > 18) and (gender == 'F') are both True. |
| True  | False | False     | (age > 18) and (gender != 'F') is False, because (gender != 'F') is False.                    |
| True  | True  | True      | (age > 18) and (gender != 'F') is False, because (gender != 'F') is False.                    |



# 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 False. |
| True  | True  | True     |  |



# Boolean/Logical Expressions

```
a = 6
b = 7
c = 42
print(1, a == 6)
print(2, a == 7)
print(3, a == 6 and b == 7)
print(4, a == 7 and b == 7)
print(5, not a == 7 and b == 7)
print(6, a == 7 or b == 7)
print(7, a == 7 or b == 6)
print(8, not (a == 7 and b == 6))
print(9, not a == 7 and b == 6)
```

## Output

```
1 True
2 False
3 True
4 False
5 True
6 True
7 False
8 True
9 False
```



# Examples

Here is a program that checks whether a number is divisible by 2 and 3, whether a number is divisible by 2 or 3, and whether a number is divisible by 2 or 3 but not both:

TestBooleanOperators

Run



# 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.

$(\text{year \% } 4 == 0 \text{ and } \text{year \% } 100 \neq 0) \text{ or } (\text{year \% } 400 == 0)$

LeapYear

Run



# 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 in any order**, the award is \$3,000.
- If one digit in the user input **matches a digit in the lottery**, the award is \$1,000.

Lottery

Run

# Conditional Operator

**if**  $x > 0$ :

$y = 1$

**else:**

$y = -1$

is equivalent to

$y = 1$  if  $x > 0$  else  $-1$

**expression1 if boolean-expression else expression2**



# Conditional Operator

```
if num % 2 == 0:  
    print(str(num) + "is even")  
else:  
    print(str(num) + "is odd");
```

**print("number is even" if (number % 2 == 0)  
else "number is odd")**



# Operator Precedence

- $+$ ,  $-$  (**Unary** plus and minus)
- $**$  (Exponentiation)
- `not`
- $*$ ,  $/$ ,  $//$ ,  $\%$  (Multiplication, division, integer division, and remainder)
- $+$ ,  $-$  (**Binary** addition and subtraction)
- $<$ ,  $<=$ ,  $>$ ,  $>=$  (Comparison)
- $==$ ,  $!=$  (Equality)
- `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

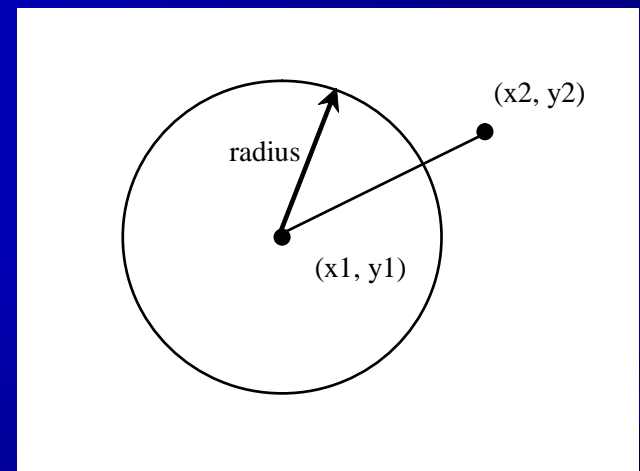
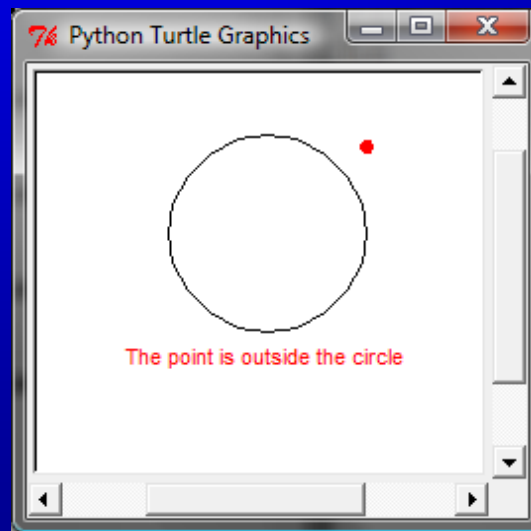
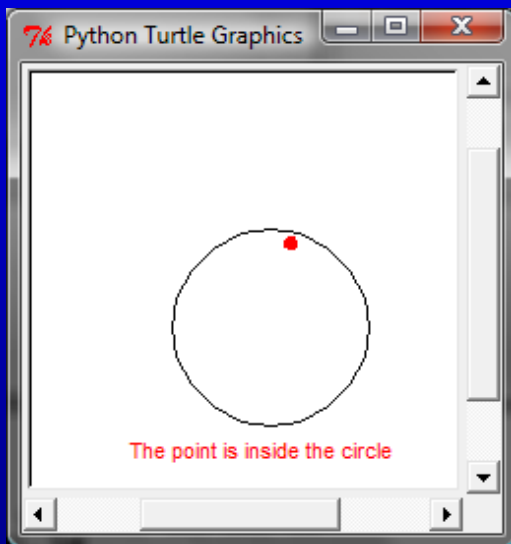
$a = b += c = 5$  is equivalent to  $a = (b += (c = 5))$



# Turtle: Location of an Object

Test whether a point is inside a circle. The program prompts the user to enter the center of a circle, the radius, and a point.

A point is in the circle if its distance to the center of the circle is less than or equal to the radius of the circle, as shown in Figure 4.7c. The formula for computing the distance is  $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ . Listing 4.11 gives the program.



[PointInCircle](#)

Run