Announcement - Midterm Exam

- Location: UU209
- 1:10pm-2:35pm (85min), Mon., 10/09/2017
- Thirty to Fourth multiple-choice questions
- Only one correct answer to each question
- Covers all materials we will have learned by midterm
- Open-book, open-note (You can bring whatever paper materials you like)
- One calculator is allowed
- Use of computers, tablets, and cellphones is prohibited (to avoid communication & cheating)

ISE 314X
Computer Programing for Engineers

Chapter 7
Decision Structures

Yong Wang
Assistant Professor
Systems Science & Industrial Engineering
Binghamton University

Objectives

- To understand `if`, `if-else`, and `if-elif-else` statements
- To understand exception handling
- To understand the `bool` data type

Simple Decisions

- We’ve viewed programs as sequences of instructions
- *Decision structures* allows the program to “choose” an appropriate course of action

Example: Temperature Warnings

- The `if` statement is used to implement the decision

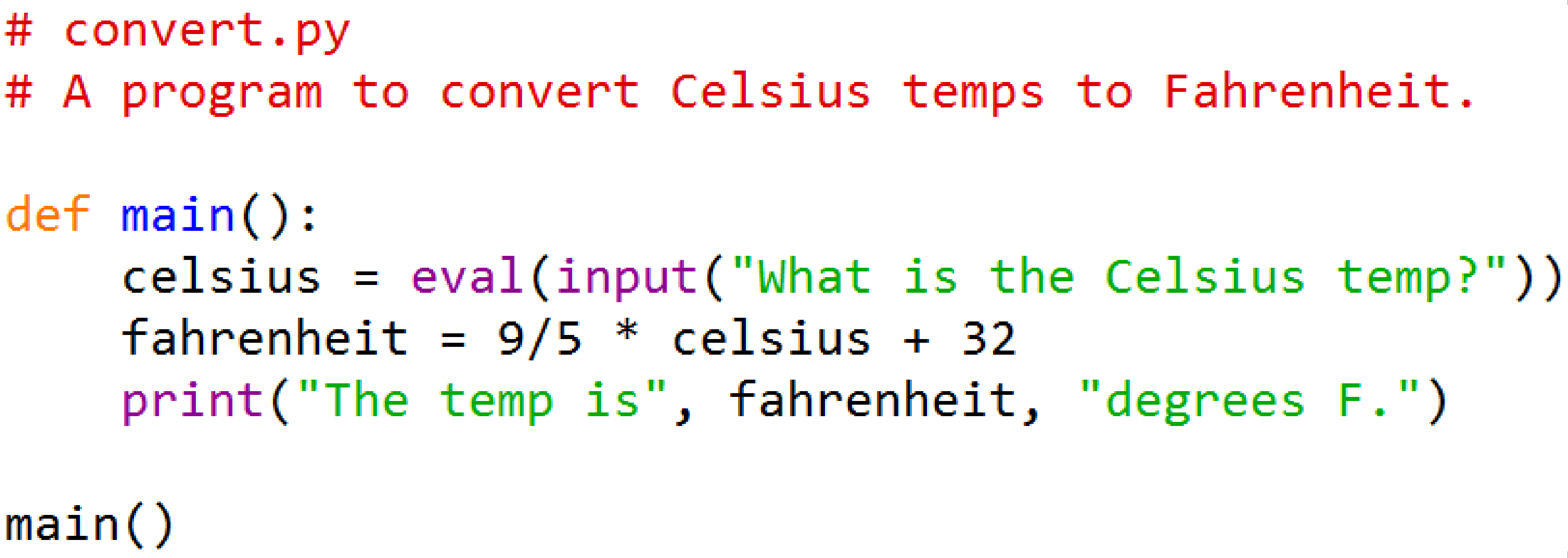
```
if <condition>:
    <body>
```

Example: Temperature Warnings

- The semantics of `if` is as follows
  - First, the condition in the heading is evaluated
  - If the condition is true, the sequence of statements in the body is executed, and then control passes to the next statement in the program
  - If the condition is false, the statements in the body are skipped, and control passes to the next statement in the program

Example: Temperature Warnings

- Celsius to Fahrenheit temperature conversion program from Chapter 2



```
# convert.py
# A program to convert Celsius temps to Fahrenheit.

def main():
    celsius = eval(input("What is the Celsius temp?"))
    fahrenheit = 9/5 * celsius + 32
    print("The temp is", fahrenheit, "degrees F.")

main()
```

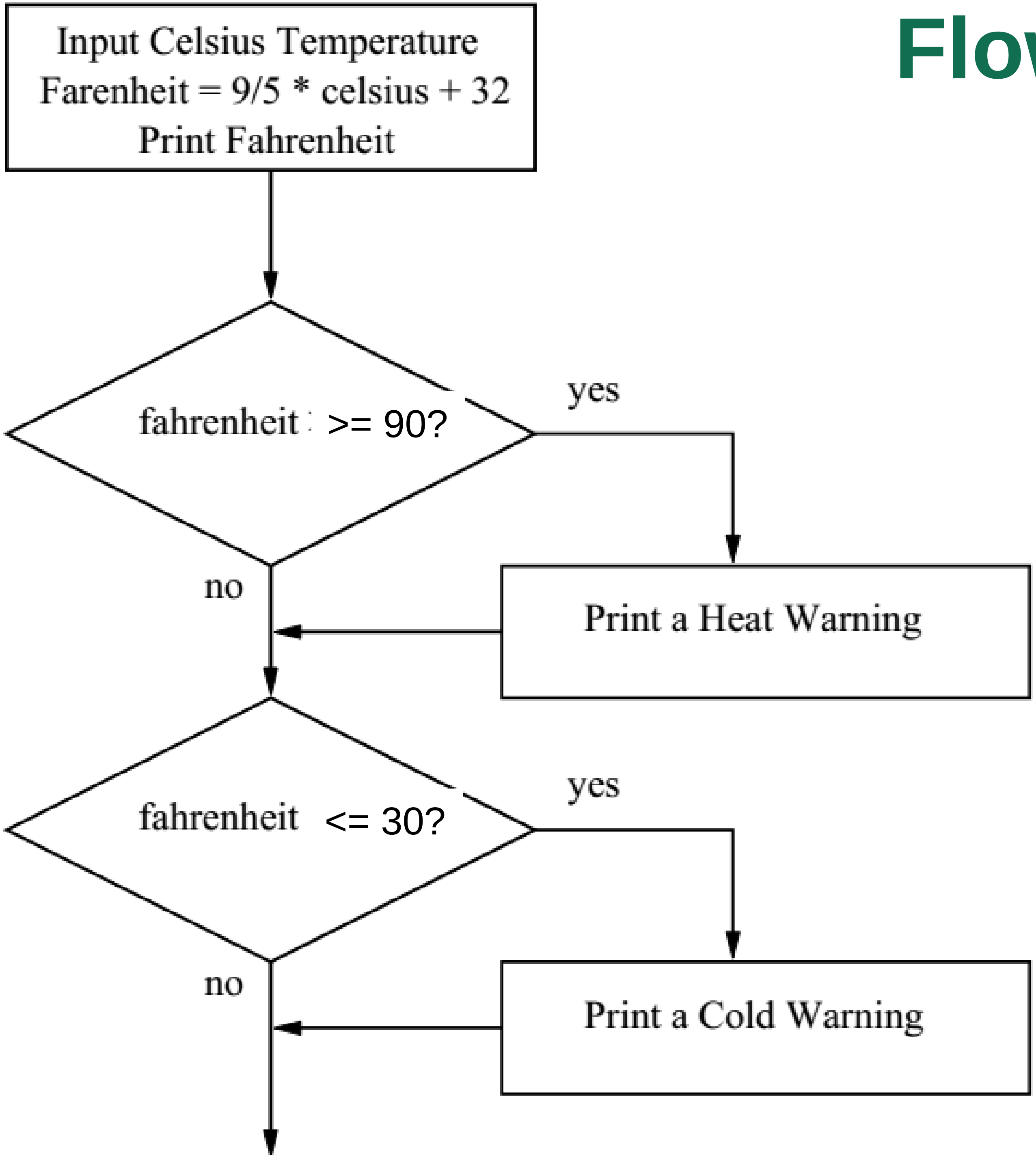
Example: Temperature Warnings

- We want to modify the program to print a warning when the weather is extreme
- Any temperature ≥ 90 degrees F will cause a hot weather warning
- Any temperature ≤ 30 degrees F will cause a cold weather warning

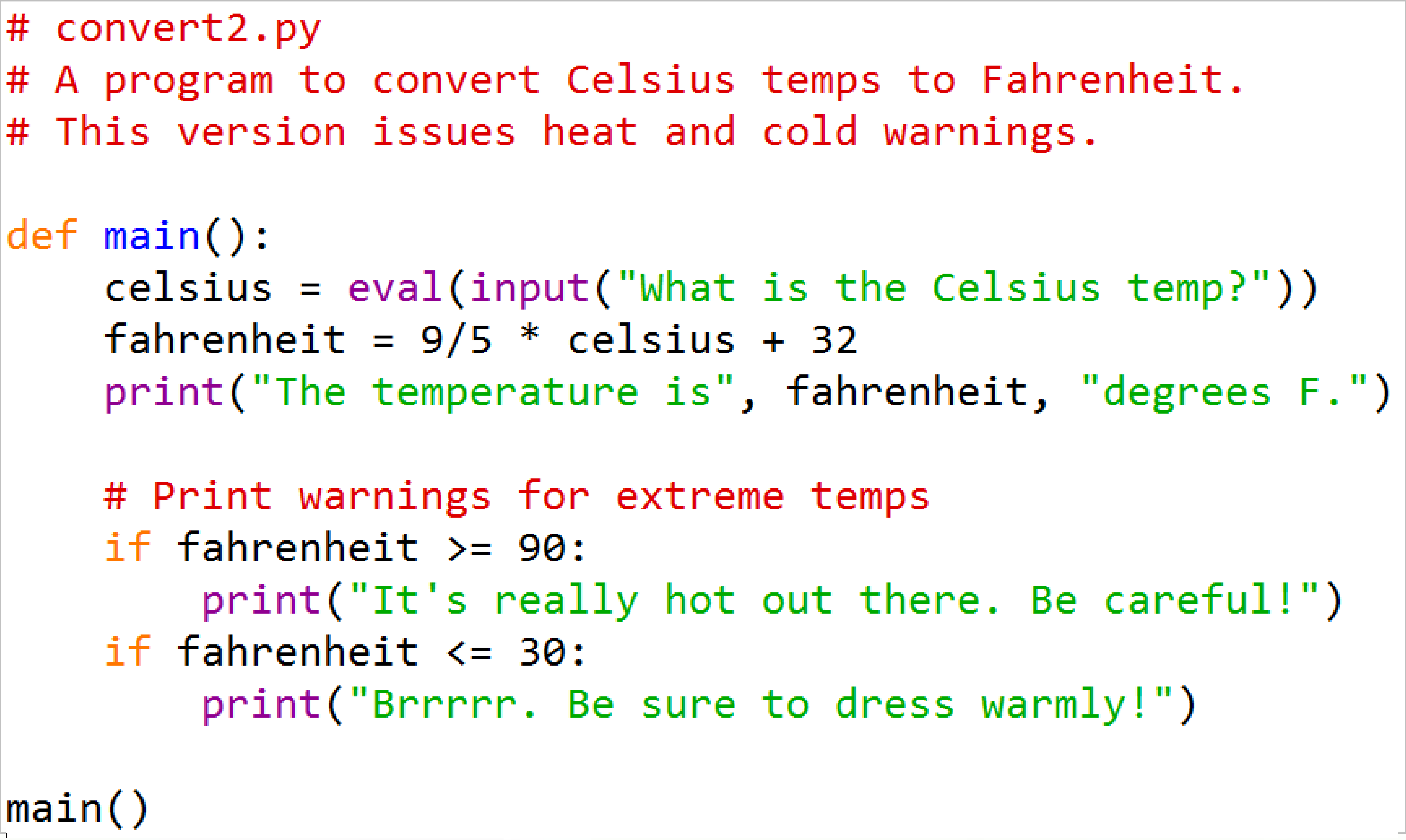
Example: Temperature Warnings

```
Input the temperature in degrees Celsius
Calculate fahrenheit as 9/5 celsius + 32
Output Fahrenheit

If fahrenheit >= 90
    print a heat warning

If fahrenheit <= 30
    print a cold warning
```

Flow Chart


Input Celsius Temperature
Farenheit = 9/5 * celsius + 32
Print Fahrenheit
fahrenheit >= 90?
yes
no
Print a Heat Warning
fahrenheit <= 30?
yes
no
Print a Cold Warning

Example: Temperature Warnings



```
# convert2.py
# A program to convert Celsius temps to Fahrenheit.
# This version issues heat and cold warnings.

def main():
    celsius = eval(input("What is the Celsius temp?"))
    fahrenheit = 9/5 * celsius + 32
    print("The temperature is", fahrenheit, "degrees F.")

    # Print warnings for extreme temps
    if fahrenheit >= 90:
        print("It's really hot out there. Be careful!")
    if fahrenheit <= 30:
        print("Brrrrr. Be sure to dress warmly!")

main()
```

Example: Temperature Warnings

- Run this program and try the following inputs

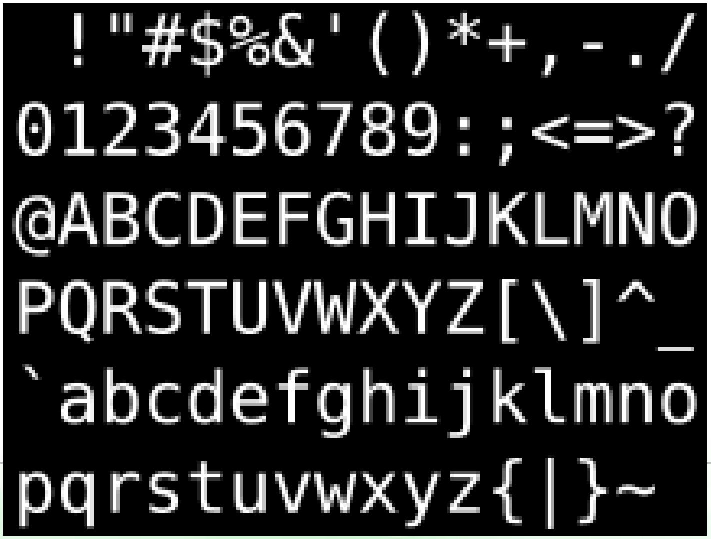
```
What is the Celsius temp?35
The temperature is 95.0 degrees F.
It's really hot out there. Be careful!

What is the Celsius temp?-5
The temperature is 23.0 degrees F.
Brrrrr. Be sure to dress warmly!
```

Forming Simple Conditions

Python condition operators	Meaning
<	Less than
<=	Less than or equal to
==	Equal to
>=	Greater than or equal to
>	Greater than
!=	Not equal to

Forming Simple Conditions

- Conditions may compare either numbers or strings
- When comparing strings, the ordering is based on the underlying [ASCII and Unicode]()
- Therefore, upper-case letters come before lower-case letters. (“Bbbb” comes before “aaaa”)



```
 !"#$%&'()*+,-./
0123456789:;<=>?
@ABCDEFGHIJKLMNO
PQRSTUVWXYZ[\]^_
`abcdefghijklmno
pqrstuvwxyz{|}~
```

Forming Simple Conditions

- Conditions are based on *Boolean* expressions, named for [George Boole]
- *True* means the condition holds. *False* means it does not hold
- Some computer languages use 1 and 0 to represent True and False, respectively

Forming Simple Conditions

```
>>> 3 < 4
True
>>> 4 * 3 < 4 + 3 #???
False
>>> "hello" == "hello"
True
>>> "Hello" < "hello"
True
>>> type(True)
<class 'bool'>
```

Example: Conditional Program Execution

- Programs or scripts are designed to be run directly
- Libraries are made to be imported and used by other programs
- Sometimes we want to create a hybrid that can be used both as a stand-alone program and as a library

Example: Conditional Program Execution

- Include the line `main()` at the bottom of the code will execute the program when it is imported
- If we do not want a program to be executed when it is imported, we can remove `main()` or make it conditional

```
if <condition>:
    main()
```

Example: Conditional Program Execution

- A special variable called `__name__` can be used to determine whether a module is imported or run directly
- __is pronounce dunder

Example: Conditional Program Execution

- **If the module is run directly**, the default value of `__name__` is `'__main__'`

```
>>> __name__
'__main__'
```

- **If a module is imported**, `__name__` represents the name of the imported module

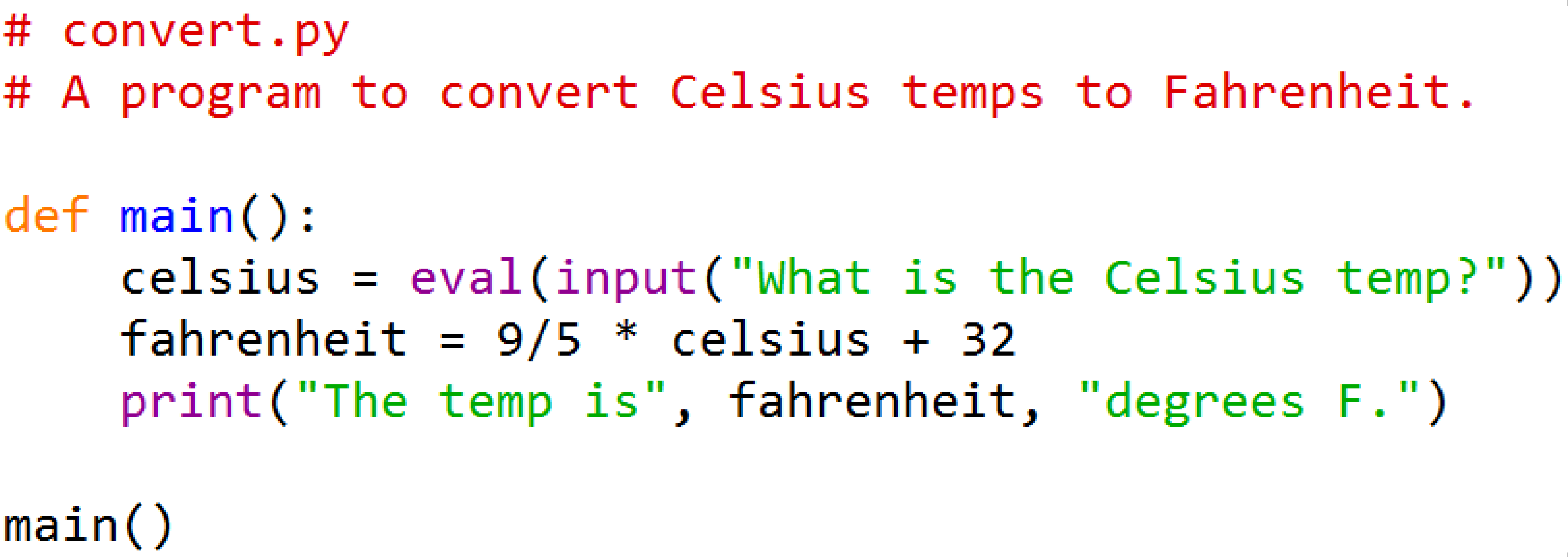
```
>>> import math
>>> math.__name__
'math'
```

Example: Conditional Program Execution

- Therefore, we can change the final line of our programs to:

```
if __name__ == '__main__':
    main()
```

Example: Temperature Warnings



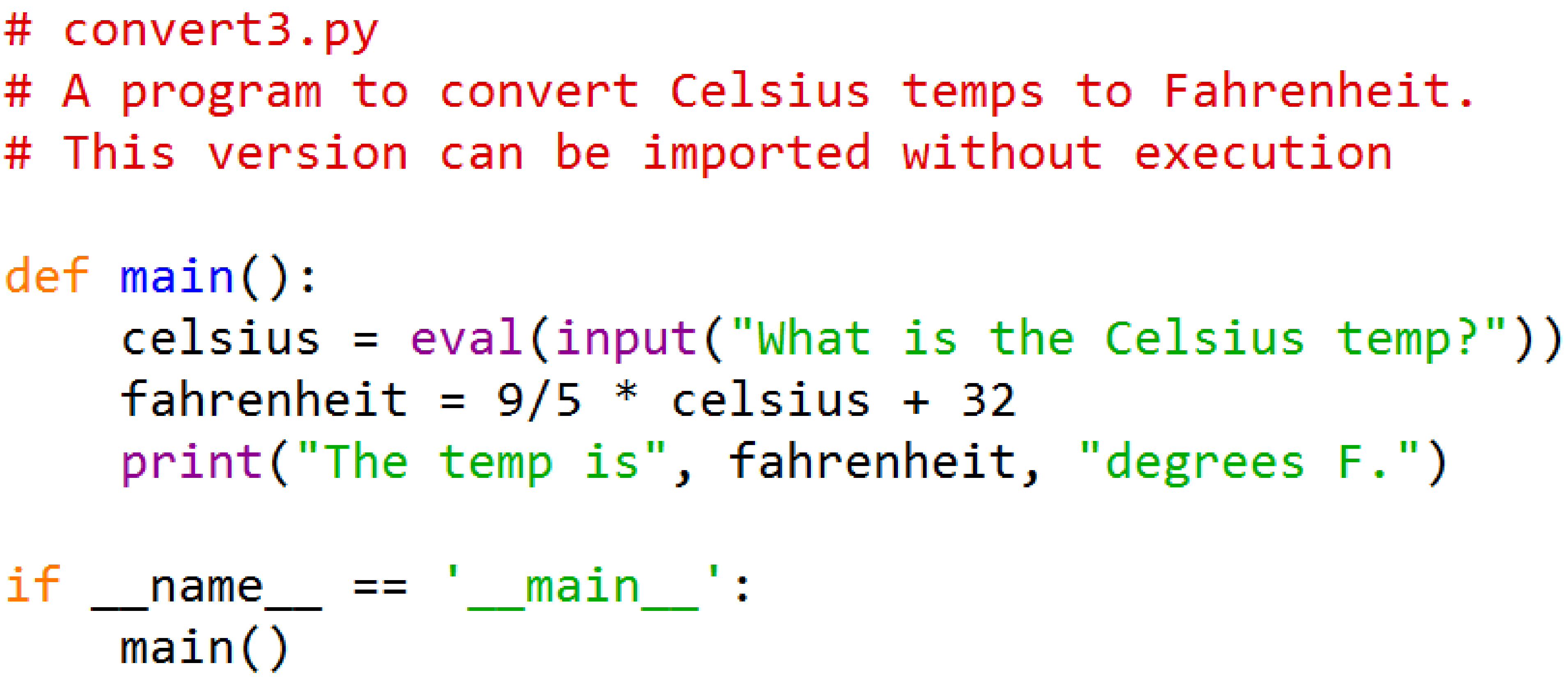
```
# convert.py
# A program to convert Celsius temps to Fahrenheit.

def main():
    celsius = eval(input("What is the Celsius temp?"))
    fahrenheit = 9/5 * celsius + 32
    print("The temp is", fahrenheit, "degrees F.")

main()
```

Example: Conditional Program Execution

- To import the program as a library

```
>>> import convert
What is the Celsius temp?25
The temp is 77.0 degrees F.
```

Example: Conditional Program Execution



```
# convert3.py
# A program to convert Celsius temps to Fahrenheit.
# This version can be imported without execution

def main():
    celsius = eval(input("What is the Celsius temp?"))
    fahrenheit = 9/5 * celsius + 32
    print("The temp is", fahrenheit, "degrees F.")

if __name__ == '__main__':
    main()
```

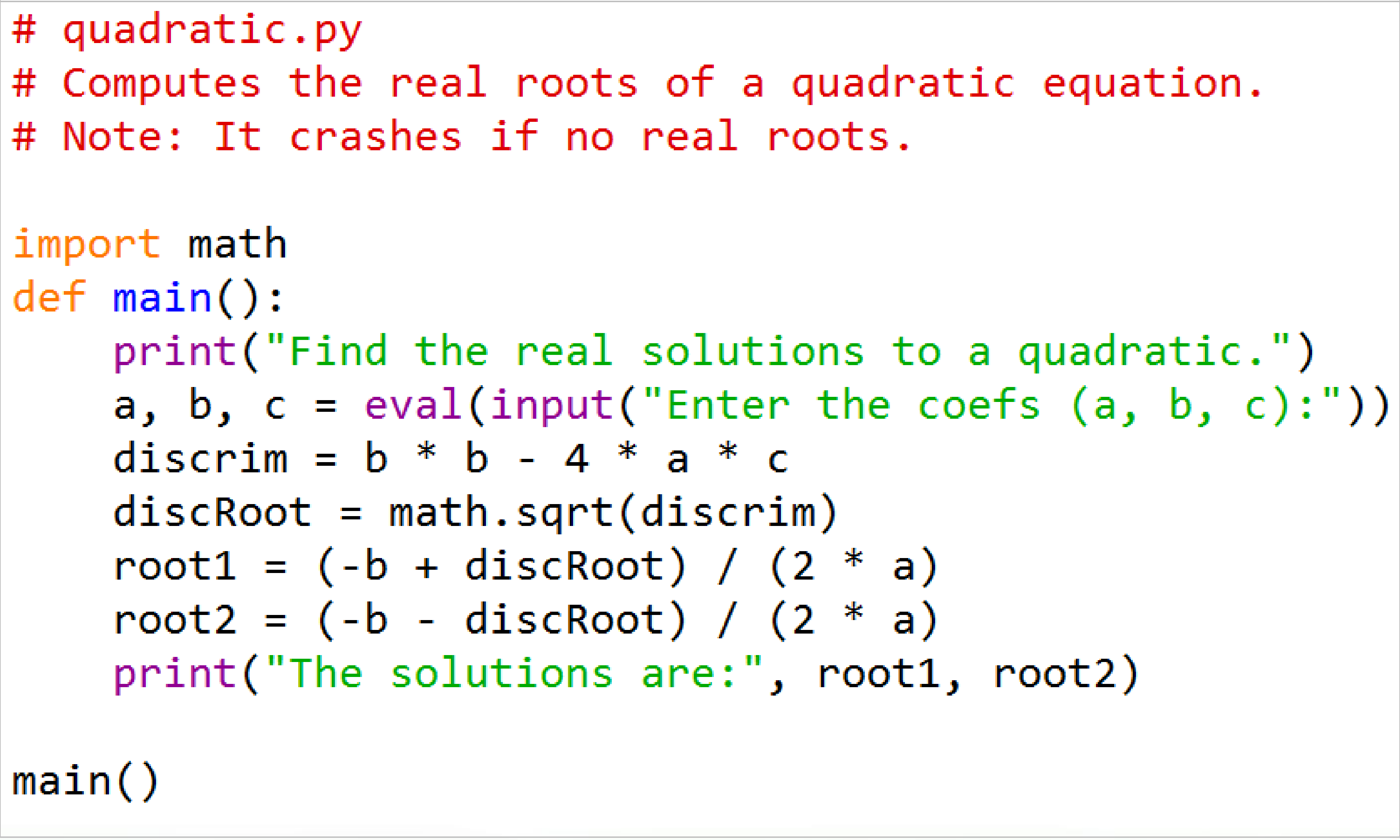
Example: Conditional Program Execution

- To import the program as a library

```
>>> import convert3
>>> convert3.main()
What is the Celsius temp?25
The temp is 77.0 degrees F.
```

- To run the program directly, use IDLE

Two-Way Decisions



```python
# quadratic.py
# Computes the real roots of a quadratic equation.
# Note: It crashes if no real roots.

import math
def main():
    print("Find the real solutions to a quadratic.")
    a, b, c = eval(input("Enter the coefs (a, b, c):"))
    discrim = b * b - 4 * a * c
    discRoot = math.sqrt(discrim)
    root1 = (-b + discRoot) / (2 * a)
    root2 = (-b - discRoot) / (2 * a)
    print("The solutions are:", root1, root2)

main()
```

Two-Way Decisions

- When b^2-$4ac$ < 0, the program crashes

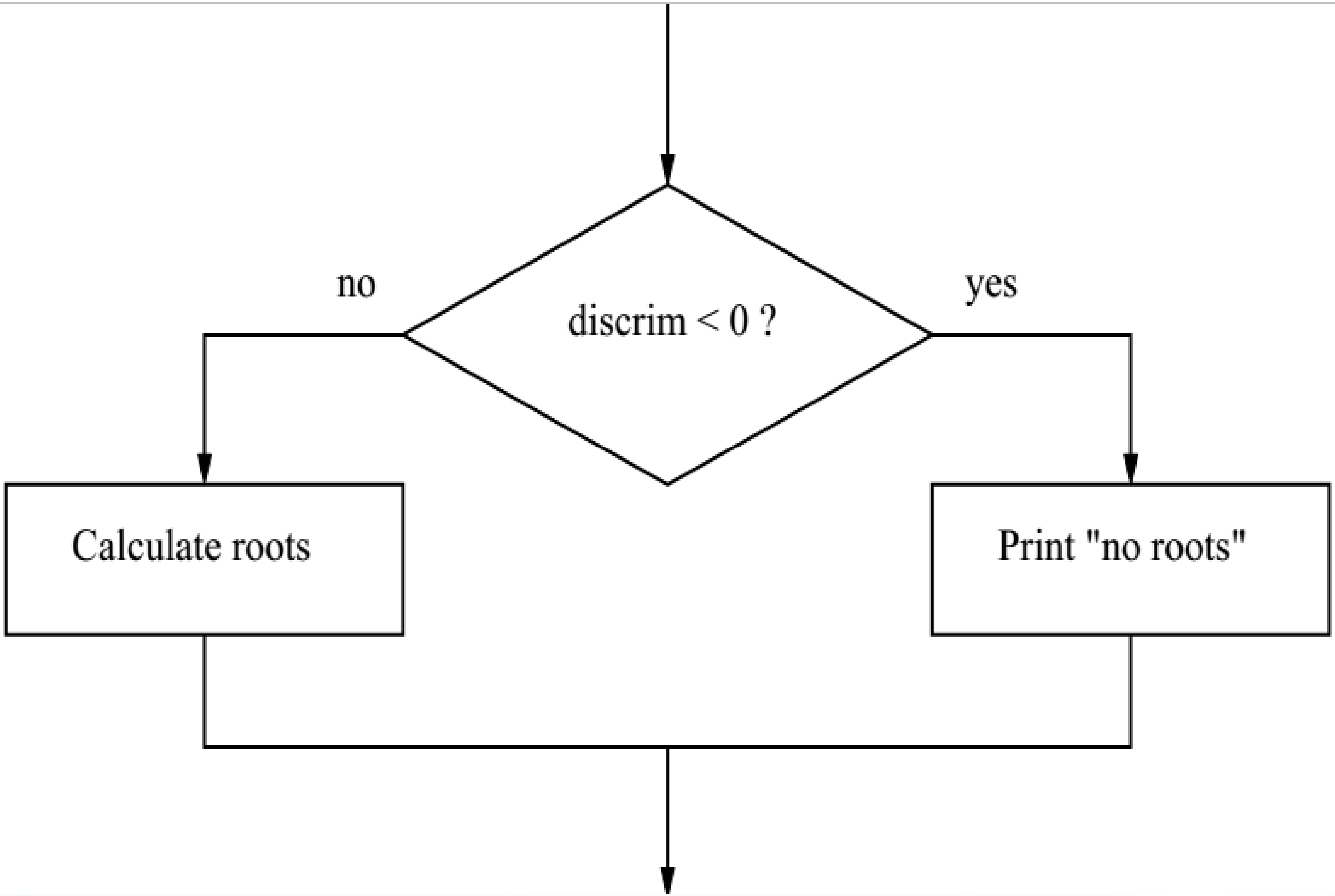
```
Finds the real solutions to a quadratic.
Enter the coefs (a, b, c): 1,2,3
Traceback (most recent call last):
  File "quadratic.py", line 14, in <module>
    main()
  File "quadratic.py", line 10, in main
    discRoot = math.sqrt(b * b - 4 * a * c)
ValueError: math domain error
```

Two-Way Decisions

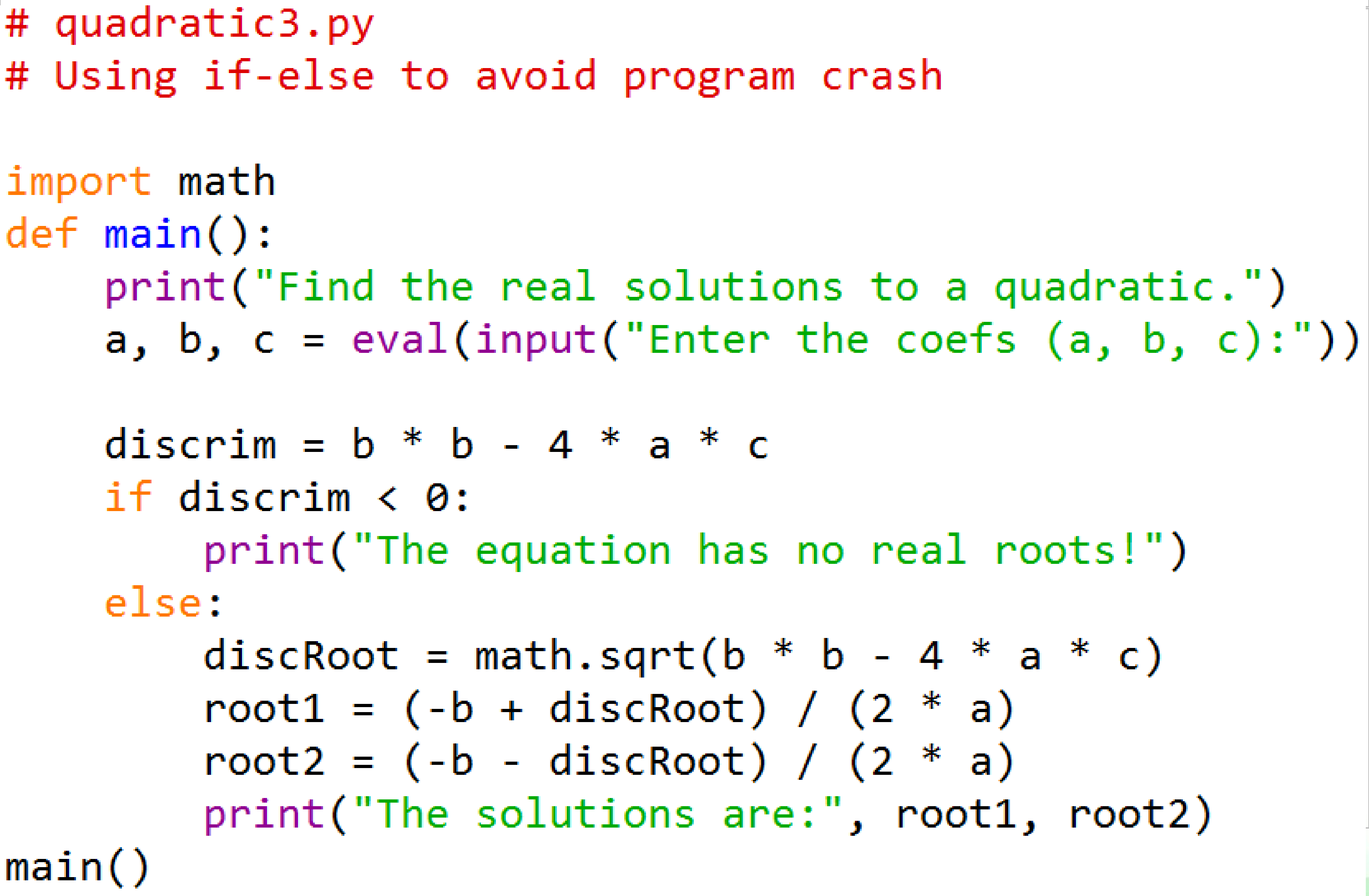
- The if-else statement

```
if <condition>:
    <statements>
else:
    <statements>
```

Two-Way Decisions



Two-Way Decisions



```python
# quadratic3.py
# Using if-else to avoid program crash

import math
def main():
    print("Find the real solutions to a quadratic.")
    a, b, c = eval(input("Enter the coefs (a, b, c):"))

    discrim = b * b - 4 * a * c
    if discrim < 0:
        print("The equation has no real roots!")
    else:
        discRoot = math.sqrt(b * b - 4 * a * c)
        root1 = (-b + discRoot) / (2 * a)
        root2 = (-b - discRoot) / (2 * a)
        print("The solutions are:", root1, root2)
main()
```

Two-Way Decisions

- Run the program

```
Find the real solutions to a quadratic.
Enter the coefs (a, b, c):1,2,3
The equation has no real roots!

Find the real solutions to a quadratic.
Enter the coefs (a, b, c):1,3,2
The solutions are: -1.0 -2.0
```

Multi-Way Decisions

- The new program is not perfect

```
Find the real solutions to a quadratic.
Enter the coefs (a,b,c):1,2,1
The solutions are: -1.0 -1.0
```

Multi-Way Decisions

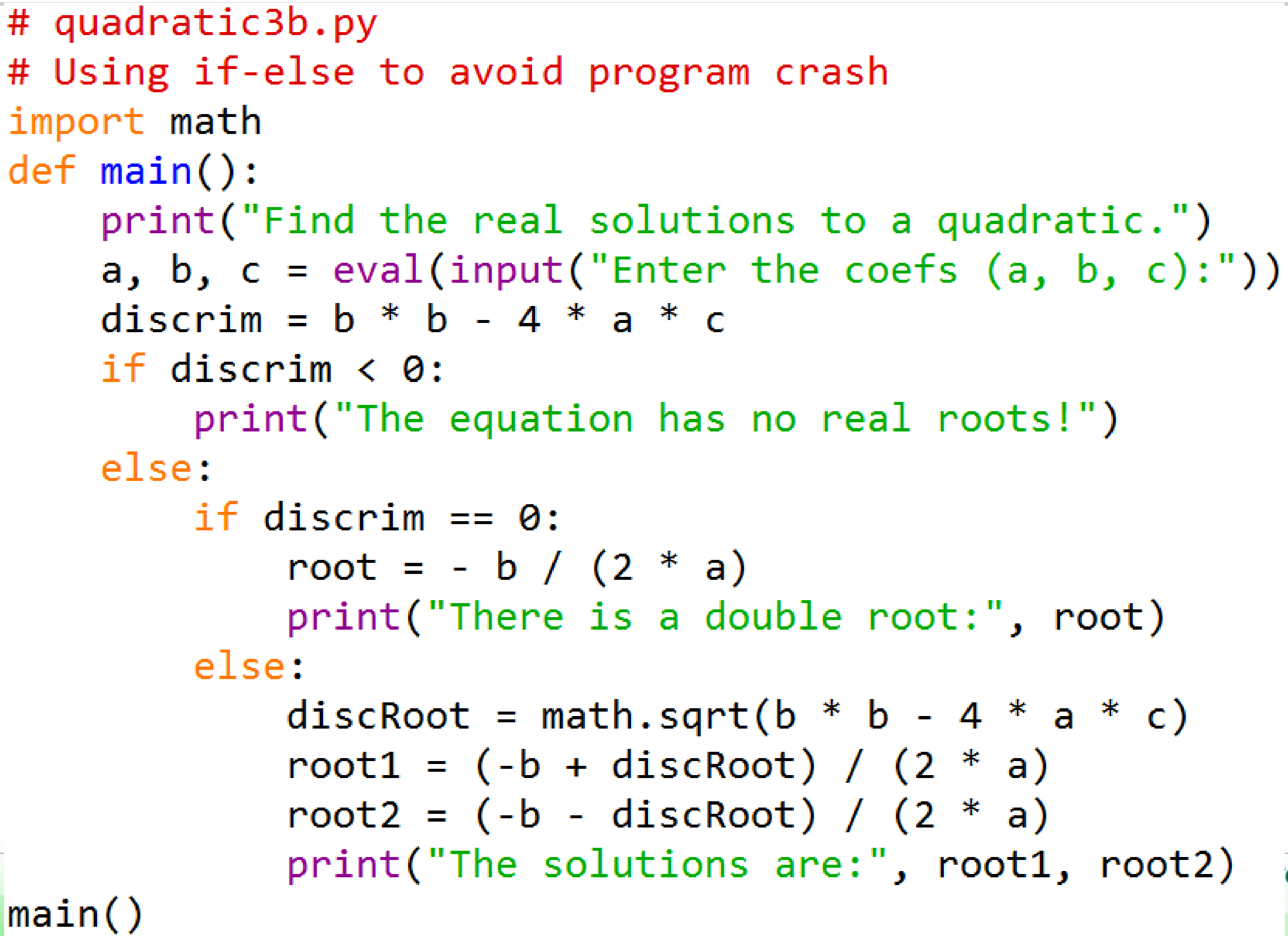
- A double root occur when the discriminant is exactly 0
- We need a three-way decision to deal with this case

Multi-Way Decisions

```
Check the value of discrim
   < 0: no roots
   = 0: a double root
   > 0: two distinct roots
```

- We can do this with
  - three if statements, or
  - two if-else statements, one inside the other. This is called *nesting*

Multi-Way Decisions



```
# quadratic3b.py
# Using if-else to avoid program crash
import math
def main():
    print("Find the real solutions to a quadratic.")
    a, b, c = eval(input("Enter the coefs (a, b, c):"))
    discrim = b * b - 4 * a * c
    if discrim < 0:
        print("The equation has no real roots!")
    else:
        if discrim == 0:
            root = - b / (2 * a)
            print("There is a double root:", root)
        else:
            discRoot = math.sqrt(b * b - 4 * a * c)
            root1 = (-b + discRoot) / (2 * a)
            root2 = (-b - discRoot) / (2 * a)
            print("The solutions are:", root1, root2)
main()
```

Multi-Way Decisions

```
Find the real solutions to a quadratic.
Enter the coefs (a, b, c):1,2,3
The equation has no real roots!

Find the real solutions to a quadratic.
Enter the coefs (a, b, c):1,3,2
The solutions are: -1.0 -2.0

Find the real solutions to a quadratic.
Enter the coefs (a,b,c):1,2,1
There is a double root: -1.0
```

Multi-Way Decisions

- Imagine if we needed to make a five-way decision using nesting
- The `if-else` statements would be nested four levels deep!
- There is a simpler way in Python: `elif`

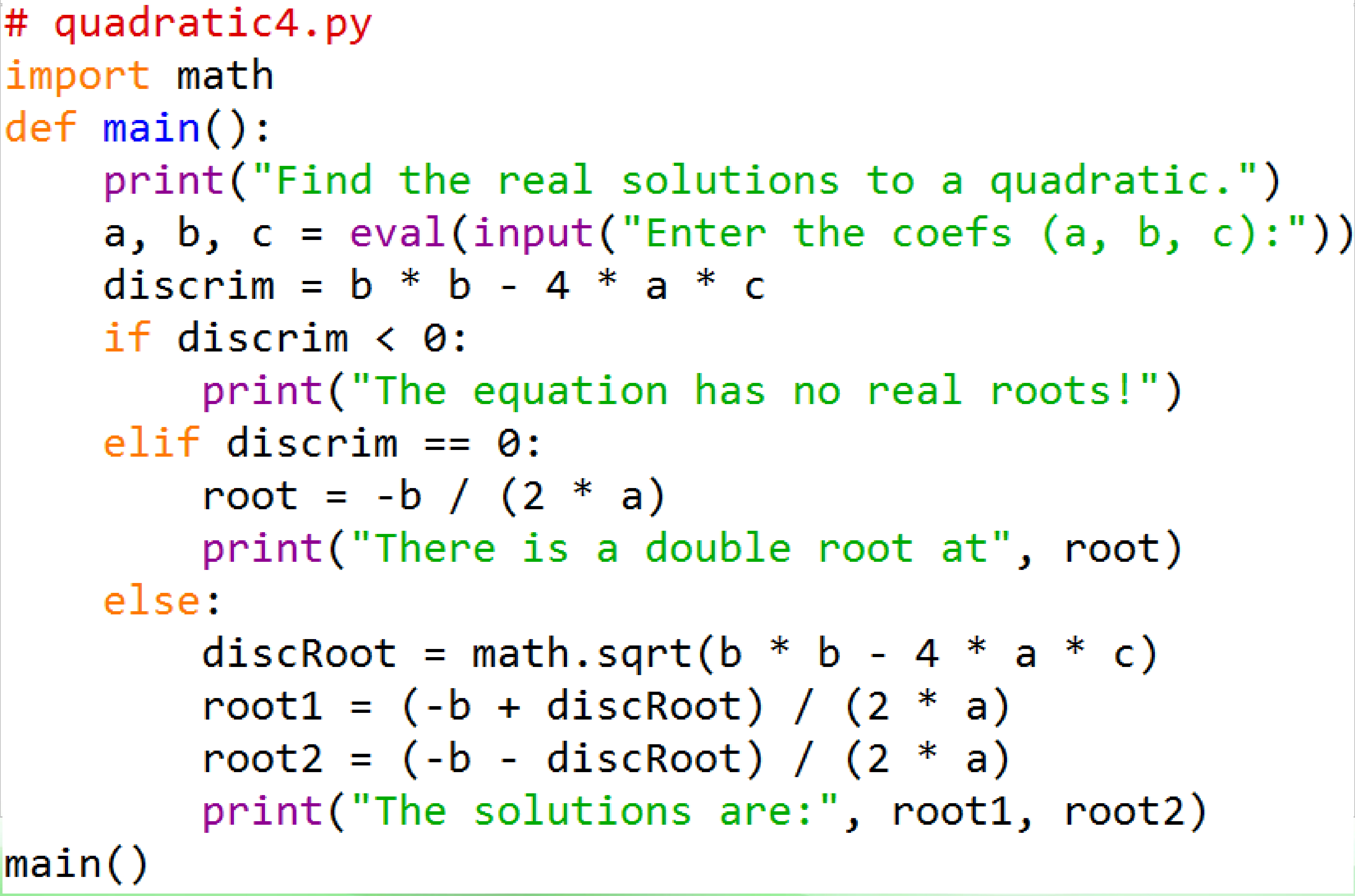
Multi-Way Decisions

```
if <condition1>:
   <case1 statements>
elif <condition2>:
   <case2 statements>
elif <condition3>:
   <case3 statements>
…
else:
   <default statements>
```

Multi-Way Decisions

- Python evaluates each condition in turn looking for the first one that is true
- If a true condition is found, the statements indented under that condition are executed, and control passes to the next statement after the entire `if-elif-else`
- If none are true, the statements under `else` are performed
- The `else` is optional. If there is no `else`, it’s possible no indented block would be executed

Multi-Way Decisions

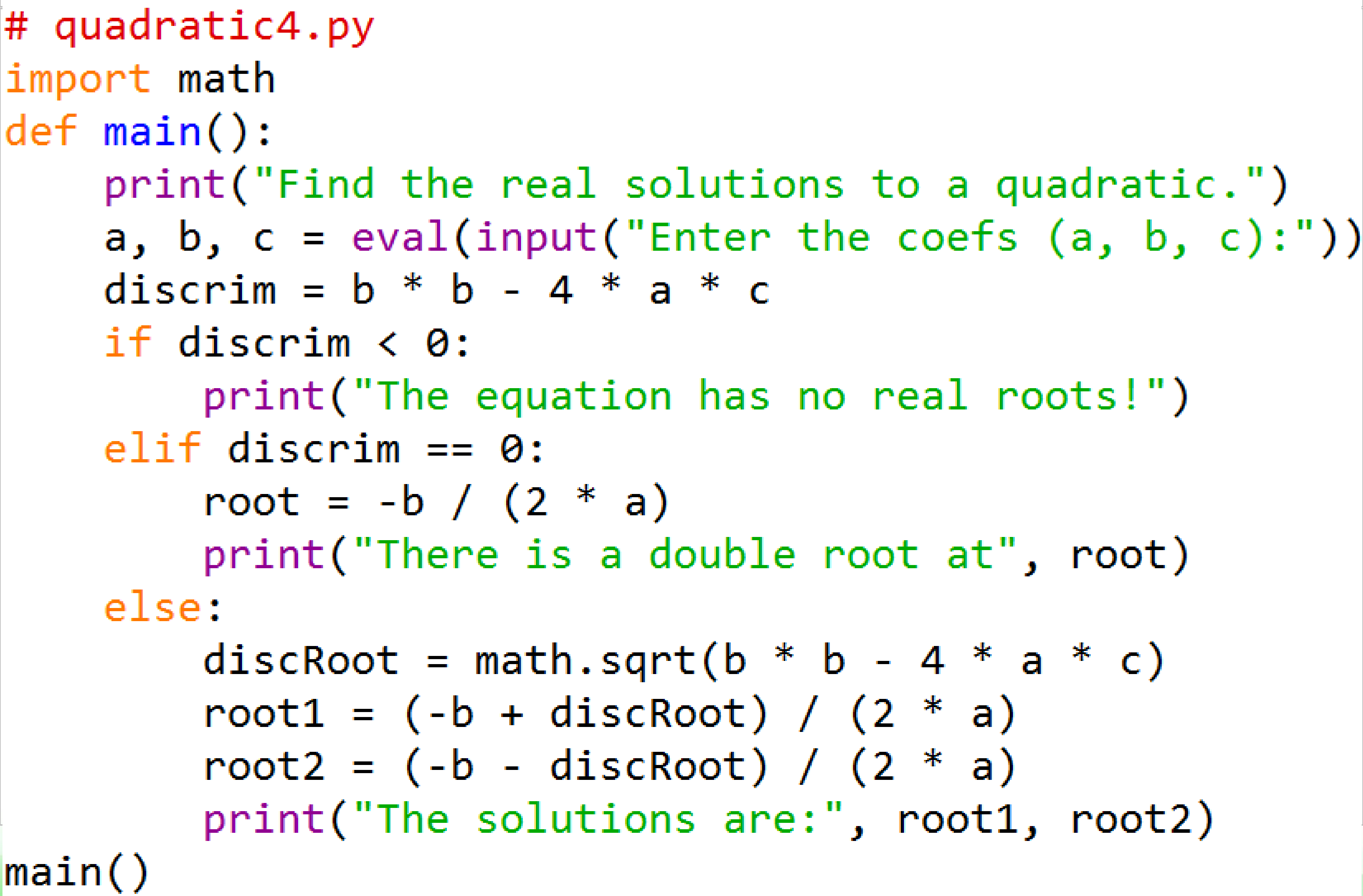


```
# quadratic4.py
import math
def main():
    print("Find the real solutions to a quadratic.")
    a, b, c = eval(input("Enter the coefs (a, b, c):"))
    discrim = b * b - 4 * a * c
    if discrim < 0:
        print("The equation has no real roots!")
    elif discrim == 0:
        root = -b / (2 * a)
        print("There is a double root at", root)
    else:
        discRoot = math.sqrt(b * b - 4 * a * c)
        root1 = (-b + discRoot) / (2 * a)
        root2 = (-b - discRoot) / (2 * a)
        print("The solutions are:", root1, root2)
main()
```

Multi-Way Decisions

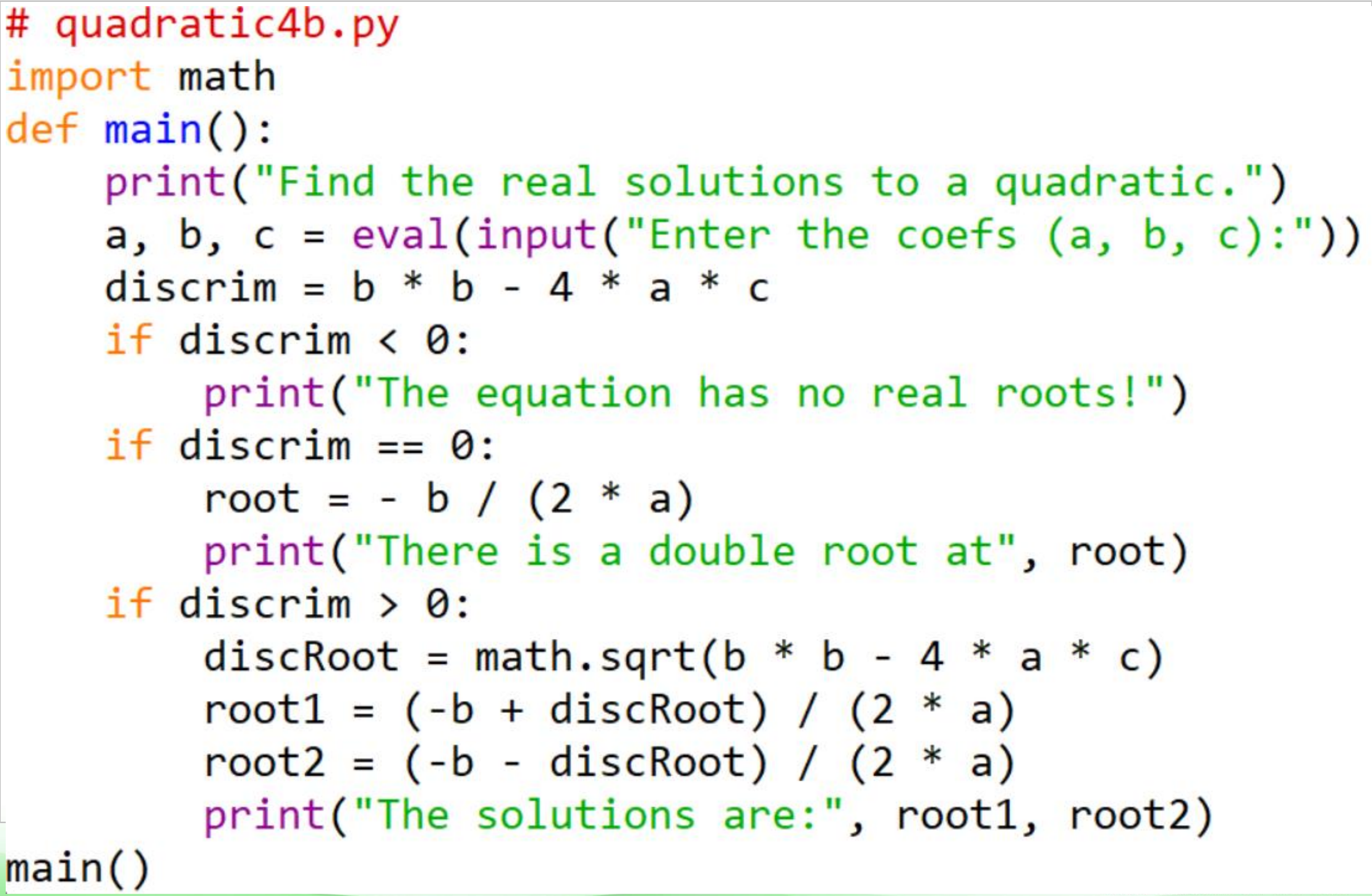
```
Find the real solutions to a quadratic.
Enter the coefs (a, b, c):1,2,3
The equation has no real roots!

Find the real solutions to a quadratic.
Enter the coefs (a, b, c):1,3,2
The solutions are: -1.0 -2.0

Find the real solutions to a quadratic.
Enter the coefs (a,b,c):1,2,1
There is a double root: -1.0
```

Modify using three if statements?



```
# quadratic4.py
import math
def main():
    print("Find the real solutions to a quadratic.")
    a, b, c = eval(input("Enter the coefs (a, b, c):"))
    discrim = b * b - 4 * a * c
    if discrim < 0:
        print("The equation has no real roots!")
    elif discrim == 0:
        root = -b / (2 * a)
        print("There is a double root at", root)
    else:
        discRoot = math.sqrt(b * b - 4 * a * c)
        root1 = (-b + discRoot) / (2 * a)
        root2 = (-b - discRoot) / (2 * a)
        print("The solutions are:", root1, root2)
main()
```

Modify using three if statements?



```python
# quadratic4b.py
import math
def main():
    print("Find the real solutions to a quadratic.")
    a, b, c = eval(input("Enter the coefs (a, b, c):"))
    discrim = b * b - 4 * a * c
    if discrim < 0:
        print("The equation has no real roots!")
    if discrim == 0:
        root = - b / (2 * a)
        print("There is a double root at", root)
    if discrim > 0:
        discRoot = math.sqrt(b * b - 4 * a * c)
        root1 = (-b + discRoot) / (2 * a)
        root2 = (-b - discRoot) / (2 * a)
        print("The solutions are:", root1, root2)
main()
```

Python IDE and Debugging

- Python IDE (Integrated Development Environment)
- PyCharm
  - – PyCharm Installation (7)
  - – PyCharm Code Writing & Debugging (11)
  - – Python Console in PyCharm (1)