DECISION STRUCTURES
CHAPTER 7.1

LEARNING OUTCOMES

- To understand the programming pattern simple decision and its implementation using a Python `if` statement.

FLOW CONTROL

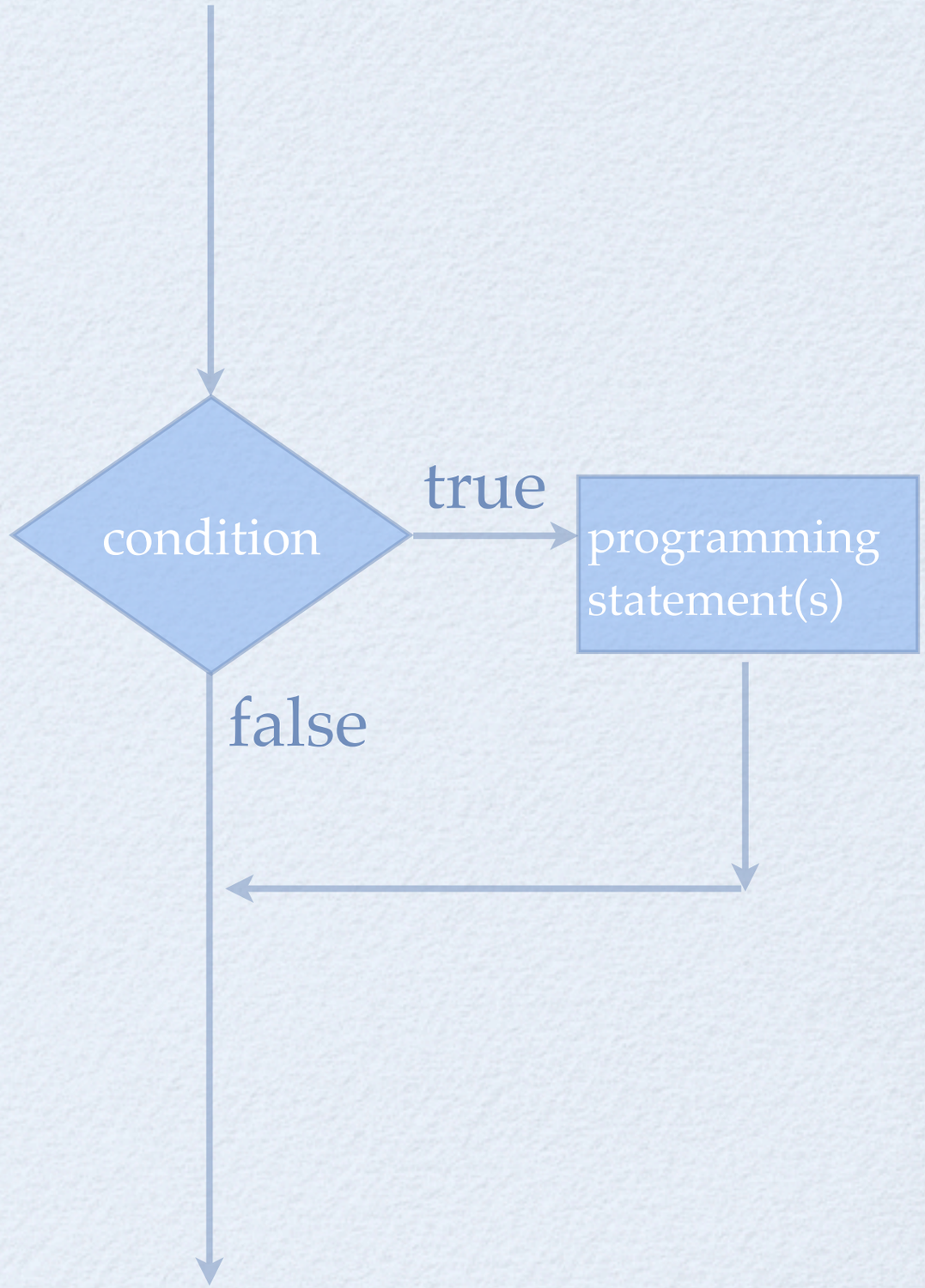
- Programming flow: the order in which program statements are executed.
- In Python, a program starts running at the first non-indented program statement that is not a function definition.
  - Usually, this is the call to main().
- Program flow then proceeds **sequentially**, waiting for one statement to finish before starting another.

PROGRAM FLOW

- So far, we have learned two methods of changing the sequential flow of program statement execution:
  - Repetition (loops): We can use these to make certain program statements repeat themselves.
  - Functions: These let us give our own names to chunks of program statements and reuse them when needed.

DECISION STRUCTURES

- Decision structures are also sometimes called selection structures, or conditionals.
- They allow us to selectively execute certain program statements, depending on whether certain conditions are true.



EXAMPLE

- Consider the following program that converts Fahrenheit temperatures into Celsius.
- Suppose we want to modify it to warn us when the temperature is too hot or too cold.

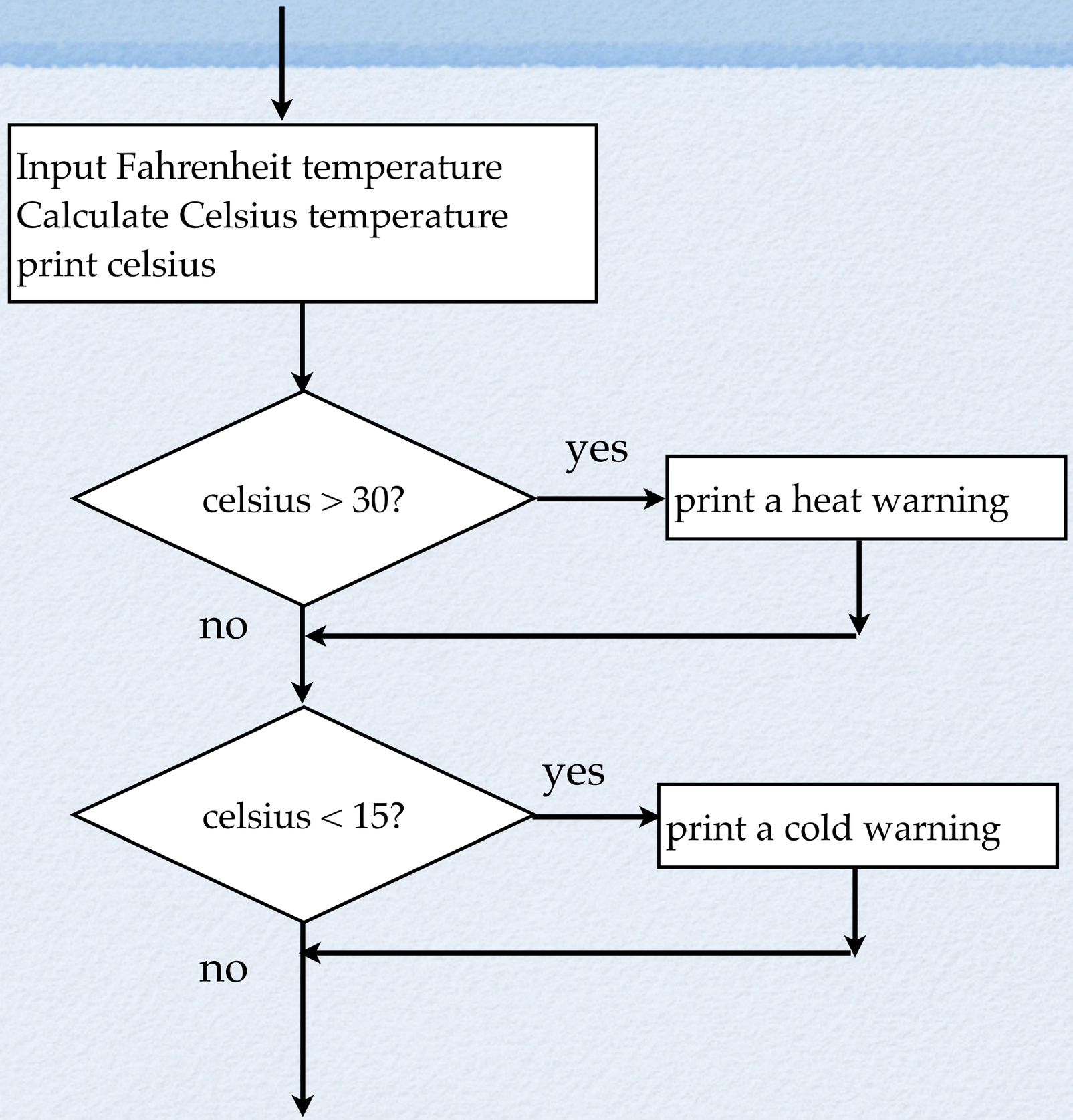
```
def main():
   fahrenheit = eval(input("Temperature in Fahrenheit?"))
   celsius = (fahrenheit - 32)*5/9
   print("Temperature is", celsius, "degrees Celsius")

main()
```

OUR ALGORITHM

- The following would be our algorithm for the modified program:

1. Input the temperature in degrees Fahrenheit (call it `fahrenheit`)
2. Calculate the temperature in Celsius (call it `celsius`) as (`fahrenheit` - 32)*5/9
3. Print `celsius`
4. If `celsius` > 30
   - Print a heat warning
5. If `celsius` < 15
   - Print a cold warning

OUR FLOWCHART



PYTHON CODE

- Python decision structure syntax looks a little like loops and functions.
- The keyword “if” precedes the condition expression, which must have a result of either true or false.
- A colon (:) follows the condition expression.
- All the statements that will be executed if the condition tests true have to be indented

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

OUR PROGRAM

```
# ConvertTemperature.py
# A program to convert Celsius to Fahrenheit
def main():
    # Get temperature and convert
    fahrenheit = eval(input("Temperature in Fahrenheit?"))
    celsius = (fahrenheit - 32)*5/9
    print("Temperature is", celsius, "degrees Celsius")

    # Print warnings for extreme weather
    if (celsius > 30):
        print("It's really hot out there!")
    if (celsius < 15):
        print("It's really cold! Get a sweater!")

main()
```

Only executed if celsius > 30

Only executed if celsius < 15

Python if statement syntax

- What we have just used here is called an **`if` statement** (because it begins with the `if` keyword).
- The syntax of an `if` statement is:

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

- <*condition*> is a Python expression that has either `True` or `False` as an "answer"
  - Recall from Lecture 3: An expression is something that has a value or that calculates a value.
- <*body*> is a block of Python program statements that are executed only if the value of the condition is `true`.

Making Conditions

- A `condition` compares the value of two expressions.

```
celsius > 30
```

- The syntax of a condition is:

<expr> <relop> <expr>

- *<relop>* stands for **relational operator**.

- Python has six relational operators.

RELATIONAL OPERATORS

- Python has six relational operators:

Python	Mathematics	Meaning
<	<	Less than
<=	≤	Less than or equal to
==	=	Equal to
>=	≥	Greater than or equal to
>	>	Greater than
!=	≠	Not equal to

BOOLEAN VALUES

- A condition is actually a special type of expression called a **Boolean expression**.
- A Boolean expression has two possible values, True (the condition holds), or False (the condition does not hold).
  - Side question: what are the possible values for an arithmetic expression?
- We can compare any two values, as long as they are of the same type (e.g. numbers compare with numbers, strings compare with strings).

Example: Conditional Program Execution

- We have been writing Python programs for a while now.
- We know that Python programs start running at the `main()` function.
- We also know that there are two ways to run a Python program:
  - Run it directly at the command prompt.

```
J:\> python test.py
```

  - Import it into the interactive shell, which automatically runs the program.

```
>>> import test
```

CONDITIONAL EXECUTION

- Sometimes, we don't want our programs to run automatically as soon as we import them.
  - Maybe we just want to try out one function in the program (we'll see more of that in a couple of chapters)
- We can use an `if` statement to perform conditional program execution:
  - If the program is run from the command prompt, go ahead and execute `main()`
  - Otherwise, just import all the program code, but don't run it.

CONDITIONAL EXECUTION

- At the end of all Python programs, after the functions are defined, there is this statement:

```
main()
```

- It tells Python to start running the main() method.

- If we want conditional program execution, we can change this line to:

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

- The Boolean condition tells Python to check if we're running the program at the command prompt -- and execute main() only if this is the case.

EXAMPLE: ERROR PREVENTION

- Problem: Write a program to solve a quadratic equation of the form:

$$ax^2 + bx + c = 0$$

- Hint: the two roots of a quadratic equation can be calculated using the formula:

$$x = \frac{-b \pm \sqrt{b^2 + 4ac}}{2a}$$

OUTPUT OF PROGRAM

```
 J:\> python quadratic.py

This program finds the solutions to a
quadratic equation, ax^2 + bx + c = 0

Please enter the coefficients (a, b, c):
1, 2, 3

The solutions are: -5.0 3.0
```

MY ALGORITHM

1. Get the user to input the three coefficients (a, b, c)

ERROR PREVENTION

- If we implement the formula as it is given, we're going to have a potential problem.

$$x = \frac{-b \pm \sqrt{b^2 + 4ac}}{2a}$$

- What is it?

ERROR PREVENTION

- We can modify our program to only calculate the roots if we can be sure that the program won't crash.

- Write down the new algorithm and the new program.

  - Hint: calculate the formula in steps, and calculate the roots of the equation only if solutions exist.

MY ALGORITHM

1. Get the user to input the three coefficients (a, b, c)

YOU NOW KNOW

- How to make simple decision structures using `if` statements in Python.