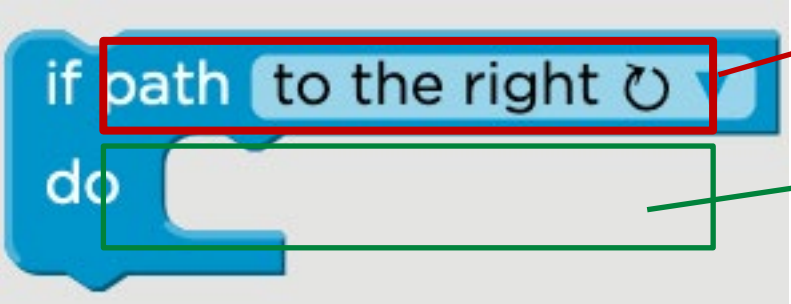
W06 LAB

Multiple Conditions

Remember the Maze Game? [recap]

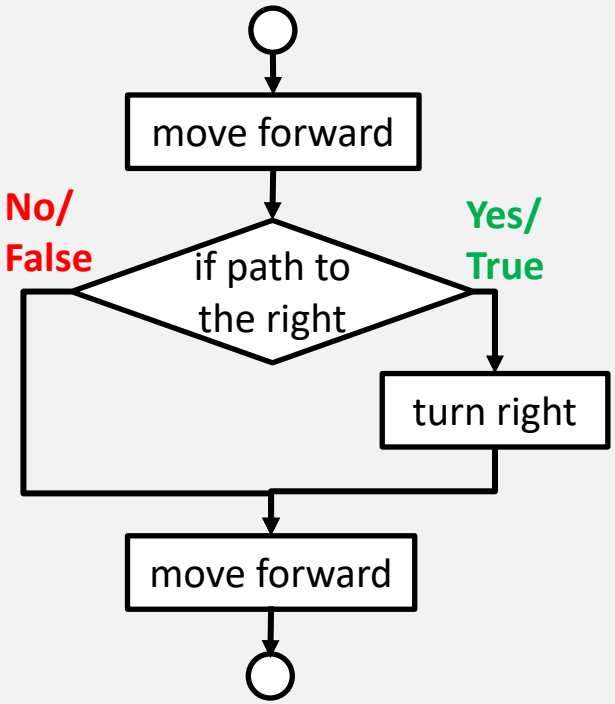


Evaluate this expression

Do this when the **expression** is True

Block-based View

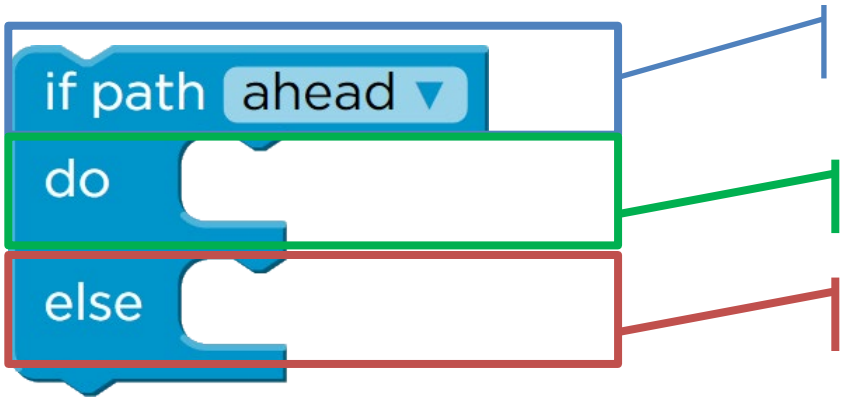
Flow Chart View



Python Code View

```
move_forward()
if path_to_the_right:
    turn_right()
move_forward()
```

Remember the Maze Game? [recap]



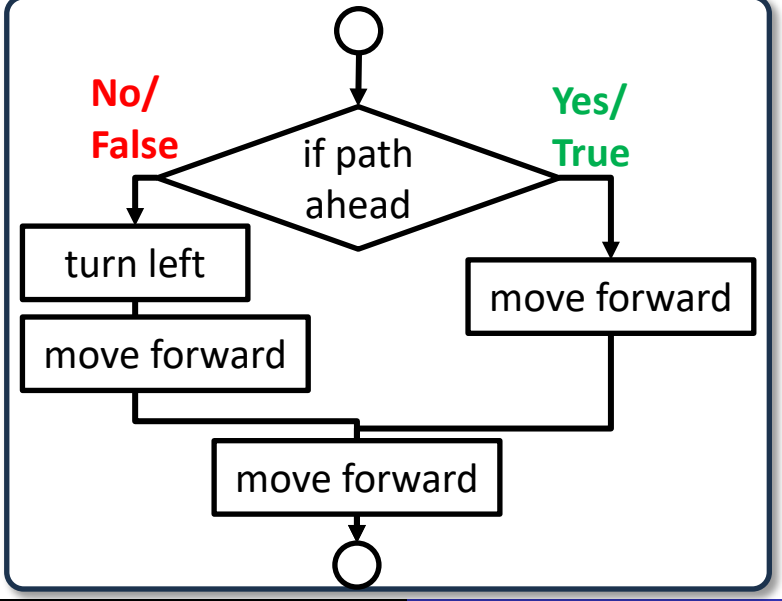
Evaluate this expression

Do this when the expression is True

Do this when the expression is False

Block-based View

Flow Chart View

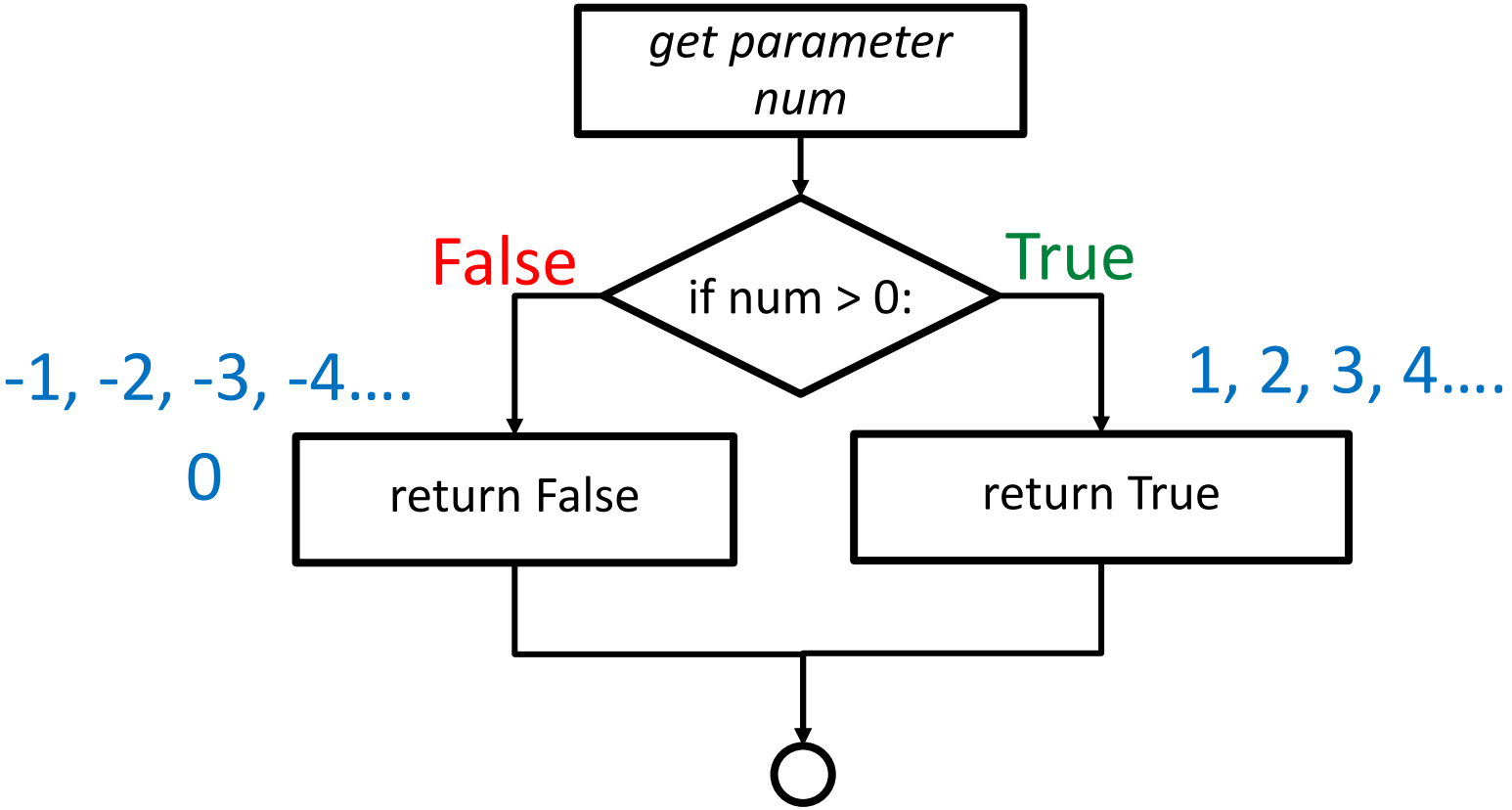


Python Code View

```
if path_ahead:
    move_forward()
else:
    turn_left()
    move_forward()
move_forward()
```

Chain Conditionals

- reconsider is_positive(num)

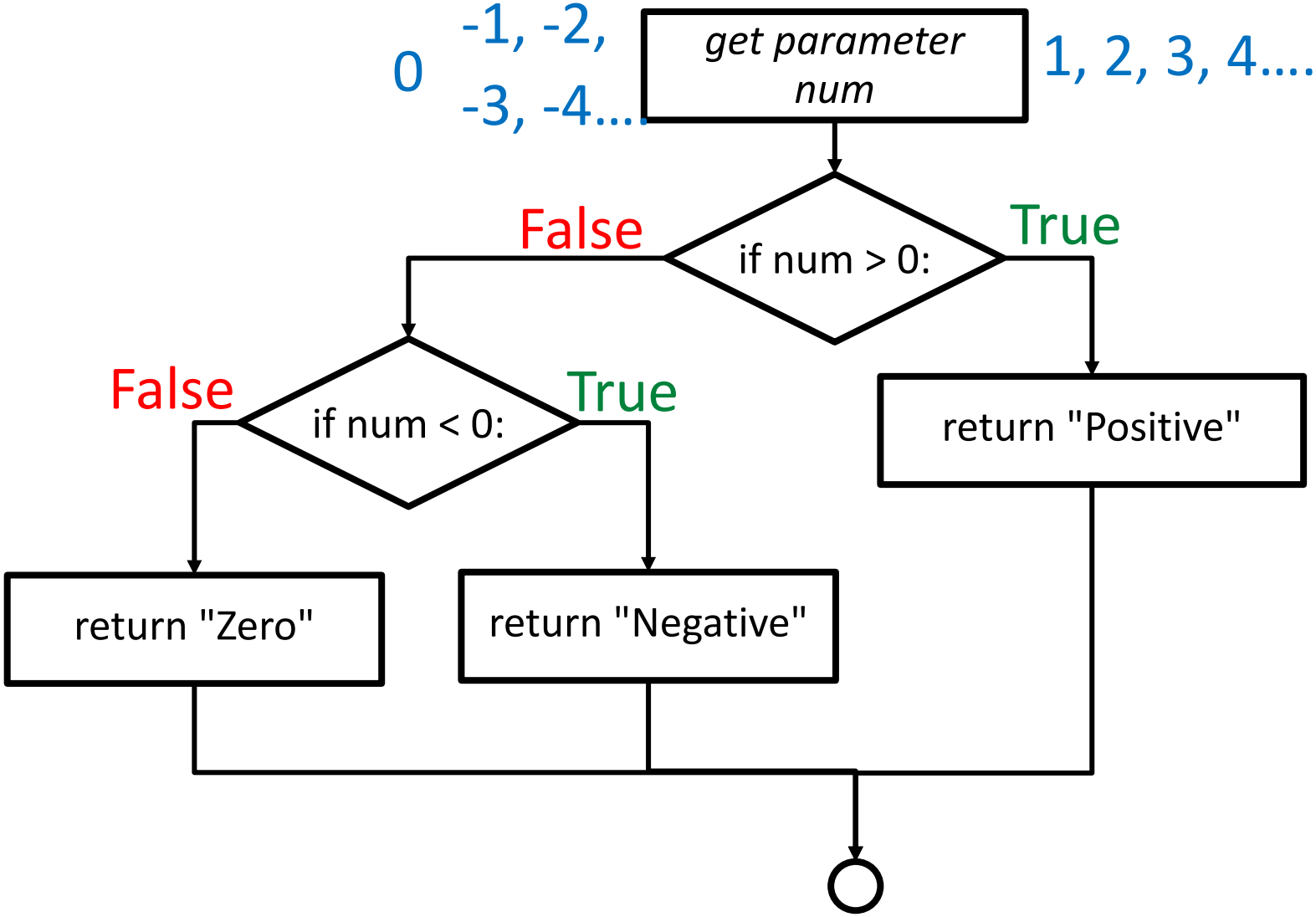


what if we want to separate positive, zero, negative number?

Multiple conditions

- the situation where more than one condition needs to be evaluated to determine the outcome or behavior of a program.
- It involves checking multiple expressions or variables against specific criteria and executing different code blocks based on the results of those conditions.

positive, zero, negative number?



Multiple Condition Problems

- Question: Discount Calculation
  - If the total purchase amount is less than $50, there is no discount.
  - If the total purchase amount is between $50 and $100 (inclusive), apply a 10% discount.
  - If the total purchase amount is greater than $100, apply a 20% discount.

Multiple Condition Problems

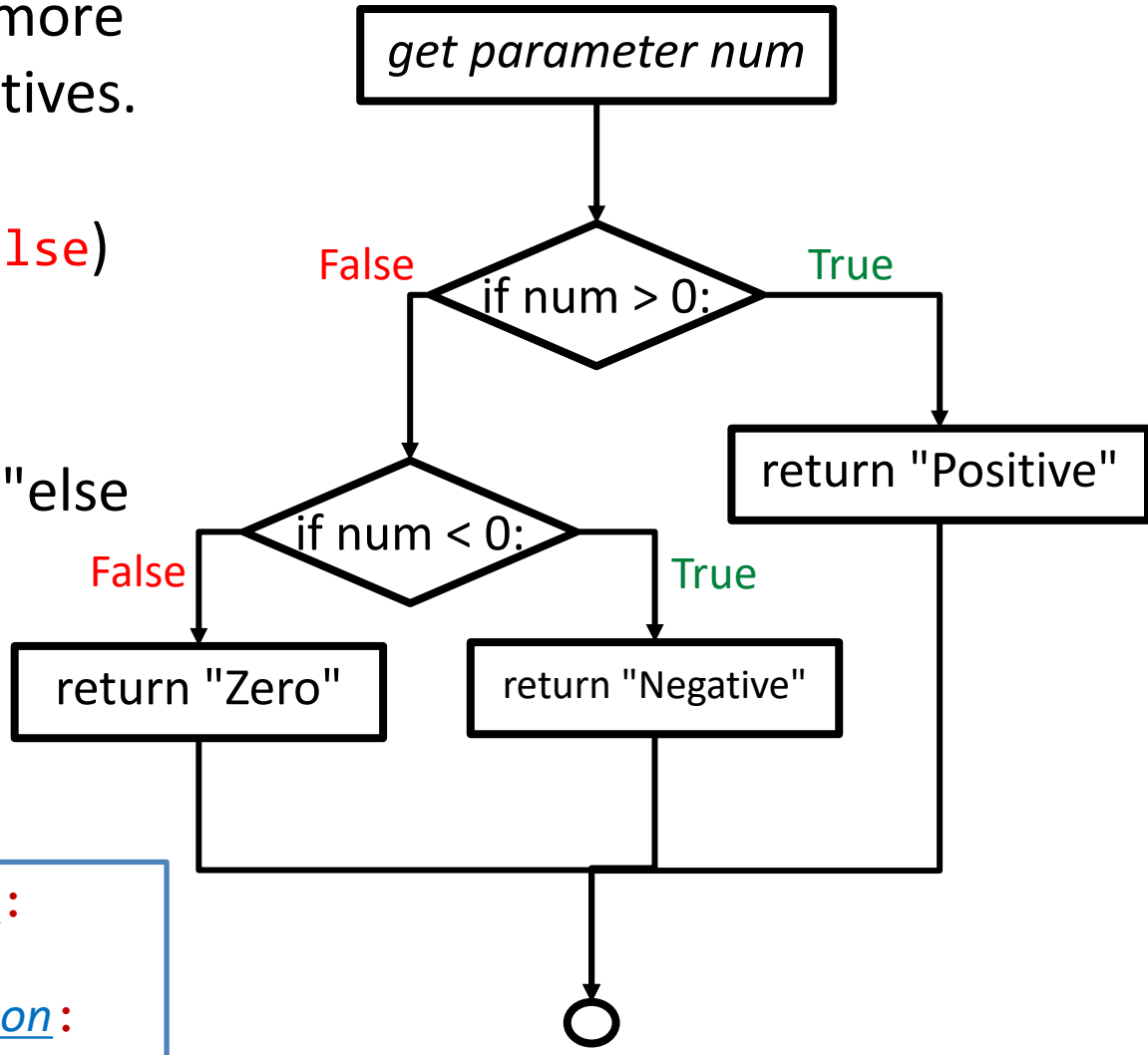
- Question: Ticket Pricing A theater has different ticket prices based on the age of the person.
  - Children (age 0-12): $5
  - Teenagers (age 13-18): $10
  - Adults (age 19-64): $15
  - Seniors (age 65 and above): $12

Multiple Condition Problems

- Question: Student Grading
  - If the score is equal to or greater than 90, it prints "Excellent!"
  - If the score is between 80 and 89 (inclusive), it prints "Good job!"
  - If the score is between 70 and 79 (inclusive), it prints "Keep it up!"
  - If the score is between 60 and 69 (inclusive), it prints "You can do better."
  - If the score is below 60, it prints "You need to study harder."

Chained Conditionals

- In some cases, there are more than two possible alternatives. We can use a chained condition (`if - elif - else`) to support this decision condition.
- elif is an abbreviation for "else if".
- from all possible options only one instruction set will be executed.

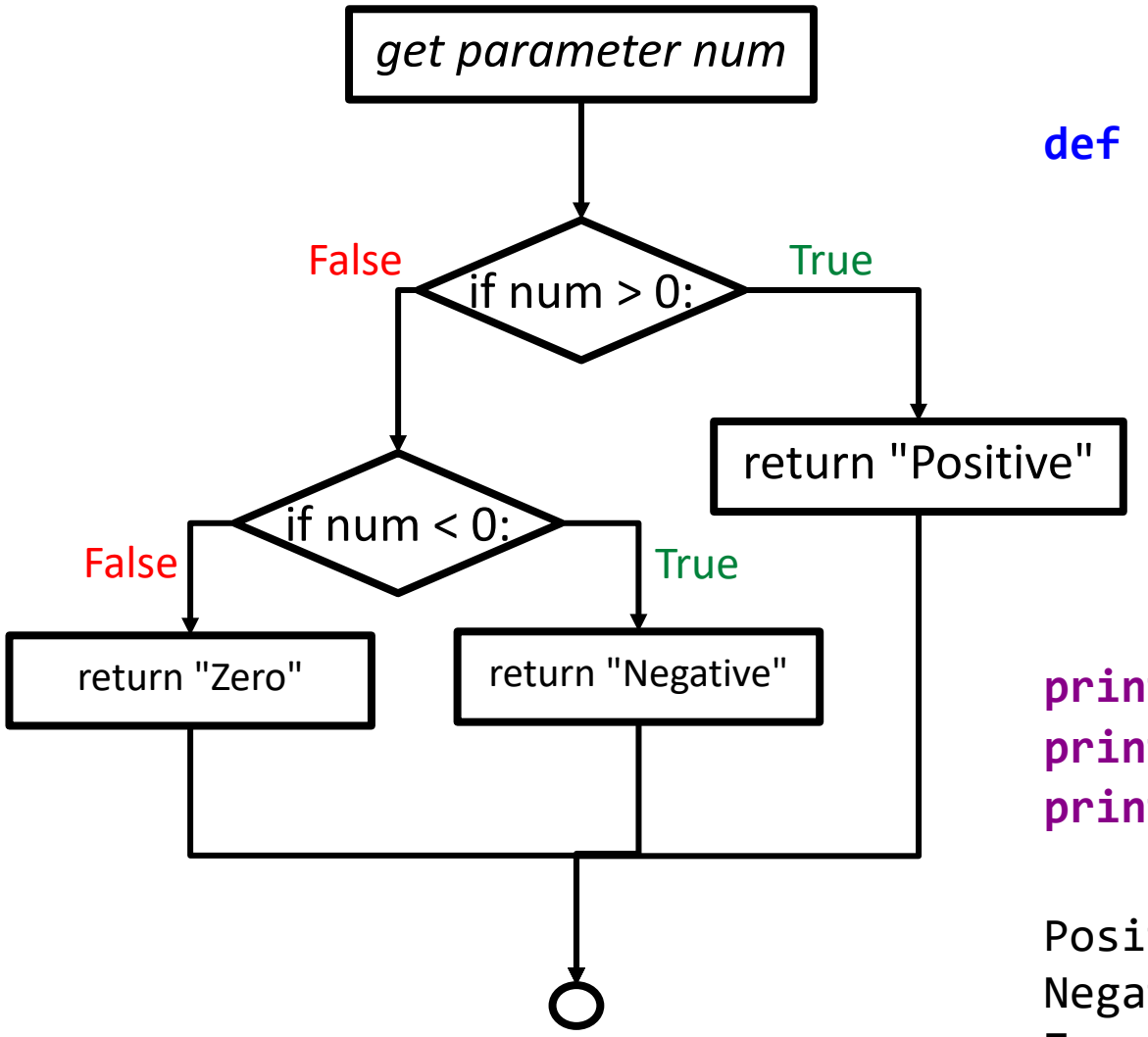



```
if Boolean expression:
    block of code
elif Boolean expression:
    block of code
else:
    block of code
```

elif (else if) Statement

- elif statement is used after if statement for further decision.
- Code block under elif statement will be executed when:
  1. The condition for the if statement (and the ones for elif statement above this one) is not met, and
  2. The condition for the current elif is met

Code vs. Flowchart




```
def int_type(num):
    if num >0:
        return "Positive"
    elif num < 0:
        return "Negative"
    else:
        return "Zero"
```

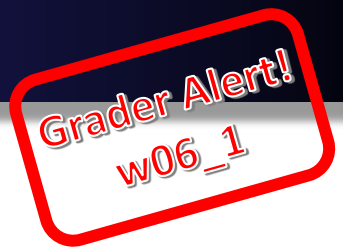
```
print(int_type(5))
print(int_type(-8))
print(int_type(0))
```

```
Positive
Negative
Zero
```

Example – Ticket Drawing

- If you randomly select a number between 1 and 99, the following outcomes are possible:
  - If the number falls between 45 and 55 (inclusive), you win the first prize.
  - If the number falls between 15 and 30 (inclusive) or between 75 and 90 (inclusive), you win the second prize.
  - For any other number, there is no prize.

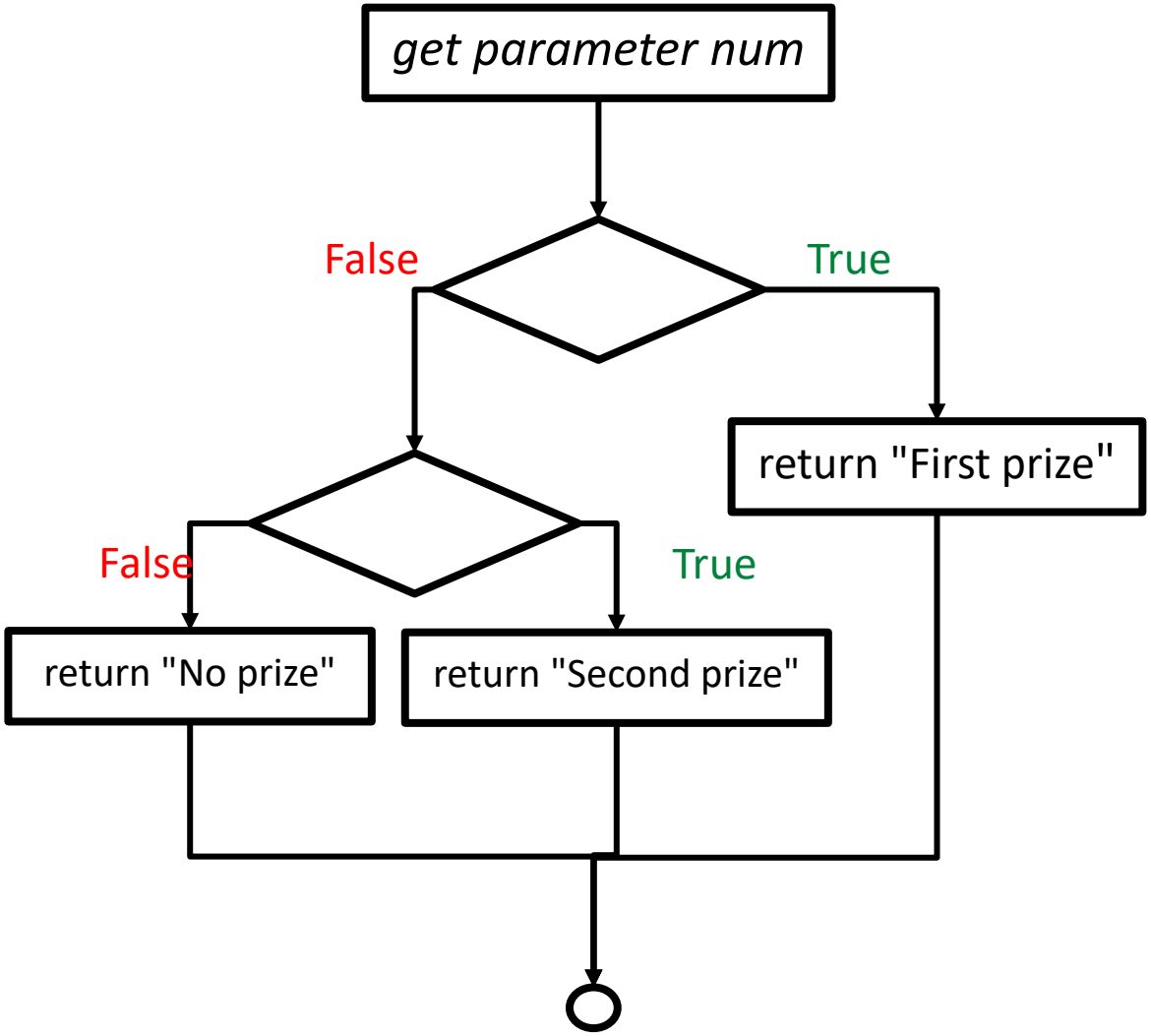
Ticket Drawing



- Break down the conditions for num
  - between 45 and 55 (inclusive) -first prize
    - Boolean expression is: 45 <= num <=55
  - between 15 and 30 (inclusive) - second prize
    - Boolean expression is:_____
  - between 75 and 90 (inclusive) - second prize
    - Boolean expression is:_____
  - other number - no prize
    - Boolean expression is:_____

Ticket Drawing

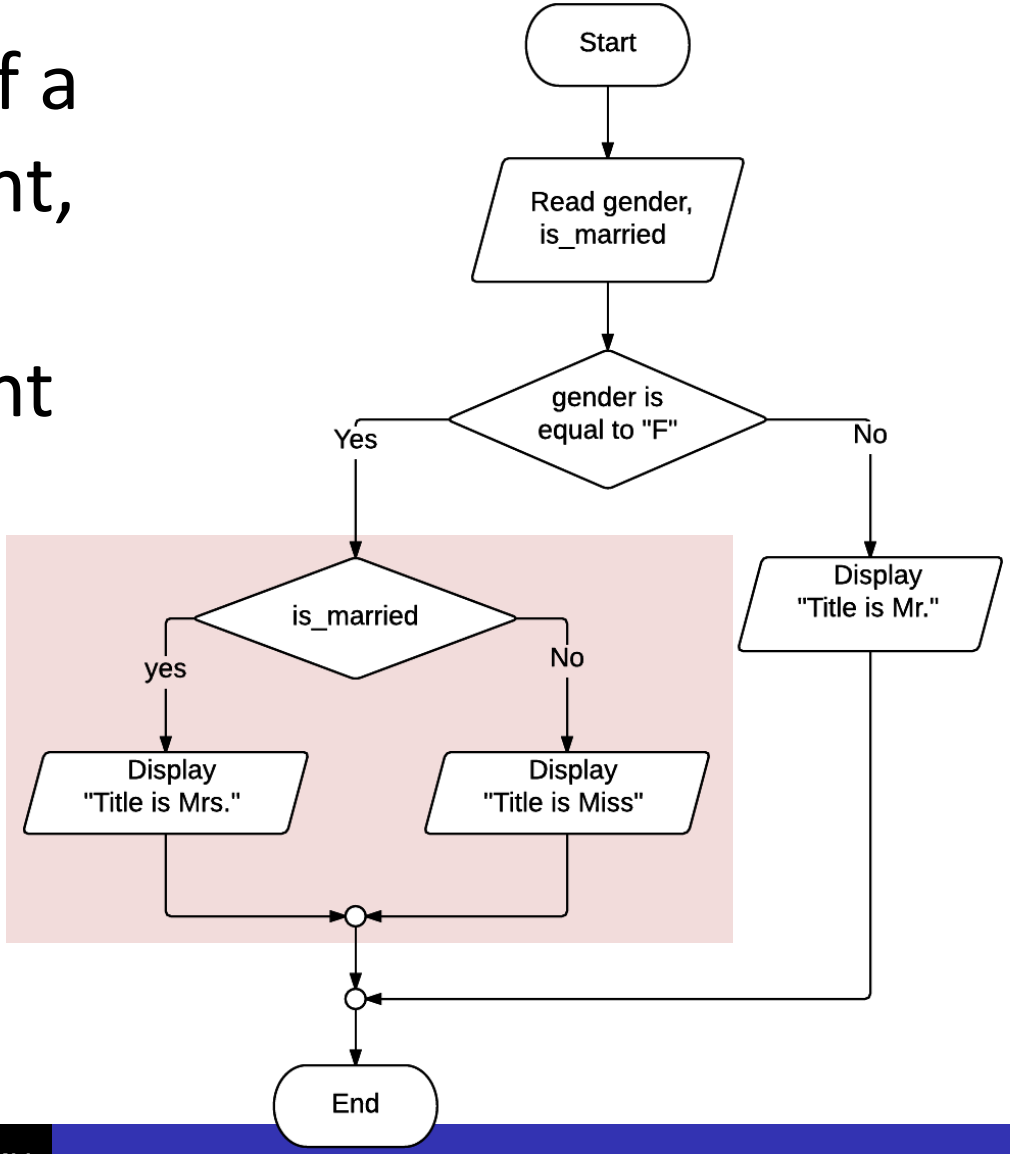
Grader Alert!
w06_1



Nested Conditionals

- Within any branch of a conditional statement, we can nest another conditional statement block.

```
if Boolean expression:
    if Boolean expression:
        block of code
    else:
        block of code
else:
    block of code
```



Nested Condition

- The program will monitor the car density of the traffic (measured in cars per kilometer, denoted as car_density) and the speed of the vehicle (measured in kilometers per hour, denoted as speed).
  - If the traffic is not heavily congested (car_density is less than or equal to 5 cars/km),
    - it is safe to drive the car at speeds up to 90 km/hr.
    - If the speed exceeds this limit, a warning should be issued.
  - However, if the traffic is congested (car_density is greater than 5 cars/km),
    - the car can only be driven at speeds up to 60 km/hr safely.
    - If the speed exceeds this limit, a warning should be given.

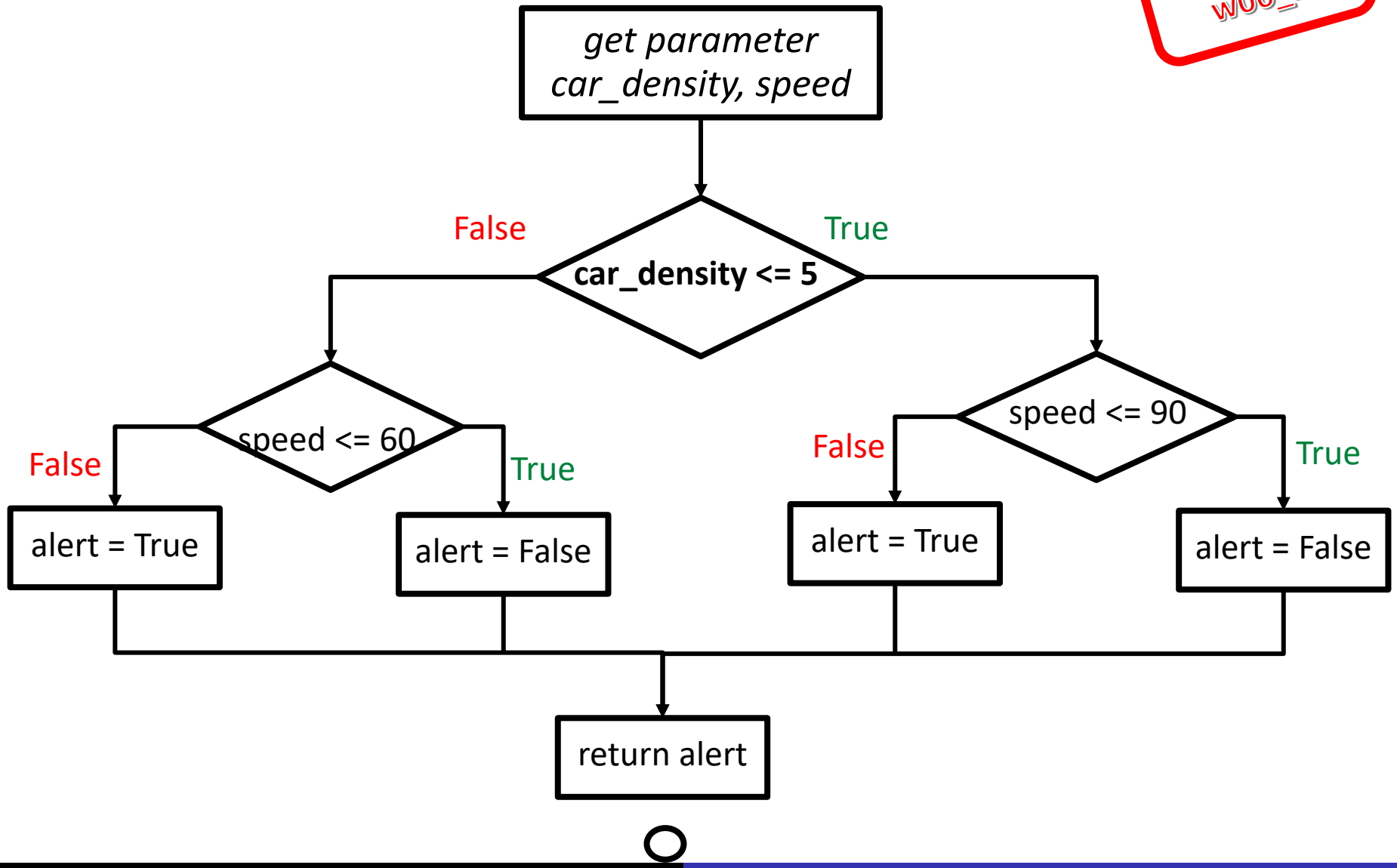
Break down the conditions

- traffic is not heavily congested (`car_density __ 5`)
  - speed ____90 km/hr. => Safe
  - otherwise => Warning

- traffic is congested (`car_density ___ 5`)
  - speed ____60 km/hr. => Safe
  - otherwise => Warning

speed_warning

Grader Alert!
w06_2



Leap Year

- A leap year is a year that has an extra day or month added to it in order to keep the calendar year aligned with the astronomical or seasonal year. To determine if a year is a leap year, the following rules can be applied:
  - If a year is not divisible by 4, it is considered a common year.
  - If a year is divisible by 4 but not by 100, it is considered a leap year.
  - If a year is divisible by both 100 and 400, it is considered a leap year.
  - For any other year that is divisible by 100 but not by 400, it is considered a common year.
  - By applying these rules, we can identify whether a given year is a leap year or not.

Example 1: Leap Year

- divisible by 4 but
  - ☑ not by 100 (2012, 2016 2020) — Case 1
  - ☑ divisible by both 100 and 400 (1600, 2000) — Case 3
  - ☒ divisible by 100 (1700, 1800, 1900) — Case 2
- STEP1: create test cases

	Test Case	output
Case 1	2012 2016 2020	YES
Case 2	1700 1800 1900	NO
Case 3	1600 2000 2400	YES
Case 4	2013 2014 2015	NO

Example 1: Leap Year

- STEP2: Consider case 1 and 4 first
  - divisible by 4 (2012, 2016 and 2020)

```
if year % 4 == 0:
    print("YES")
else:
    print("NO")
```

	Test Case	output
Case 1	2012 2016 2020	YES
Case 2	1700 1800 1900	NO
Case 3	1600 2000 2400	YES
Case 4	2013 2014 2015	NO

Example 1: Leap Year [2]

- STEP3: Consider Case 2
  - divisible by 4 but
    - ☒ divisible by 100 (1700, 1800, 1900)

```
if year % 4 == 0:
    print("YES")


else:
    print("NO")
```

	Test Case	output
Case 1	2012 2016 2020	YES
Case 2	1700 1800 1900	NO
Case 3	1600 2000 2400	YES
Case 4	2013 2014 2015	NO

Example 1: Leap Year [3]

- STEP3: Consider Case 2
  - divisible by 4 but
    - ☒ divisible by 100 (1700, 1800, 1900)

```
if year % 4 == 0:    //nested under case 1
    if year % 100 == 0:
        print("NO")
    else:
        print("YES")

else:
    print("NO")
```

	Test Case	output
Case 1	2012 2016 2020	YES
Case 2	1700 1800 1900	NO
Case 3	1600 2000 2400	YES
Case 4	2013 2014 2015	NO

Example 1: Leap Year [4]

- STEP3: Consider Case 3
  - divisible by 4 but
    - ☒ divisible by 100 (1700, 1800, 1900)
    - ☑ divisible by both 100 and 400 (1600, 2000)

```
if year % 4 == 0:
    if year % 100 == 0:
        print("NO")


    else:
        print("YES")

else:
    print("NO")
```

	Test Case	output
Case 1	2012 2016 2020	YES
Case 2	1700 1800 1900	NO
Case 3	1600 2000 2400	YES
Case 4	2013 2014 2015	NO

Example 1: Leap Year [5]

- STEP3: Consider Case 3
  - divisible by 4 but
    - ☒ divisible by 100 (1700, 1800, 1900)
    - ☑ divisible by both 100 and 400 (1600, 2000)

```
if year % 4 == 0:
    if year % 100 == 0:
        print("NO")
    elif year % 400 == 0:
        print("YES")
    else:
        print("YES")

else:
    print("NO")
```

	Test Case	output
Case 1	2012 2016 2020	YES
Case 2	1700 1800 1900	NO
Case 3	1600 2000 2400	YES
Case 4	2013 2014 2015	NO

Example 1: Leap Year [6]

- STEP5: Testing

```
if year % 4 == 0:
  if year % 100 == 0:
    print("NO")
  elif year % 400 == 0:
    print("YES")
  else:
    print("YES")

else:
  print("NO")
```

Solution: Swap the condition position.

	Test Case	output
Case 1	2012 2016 2020	YES
Case 2	1700 1800 1900	NO
Case 3	1600 2000 2400	YES
Case 4	2013 2014 2015	NO

- Year = 2400 output = ?
- Why?
- How to fix the bug?

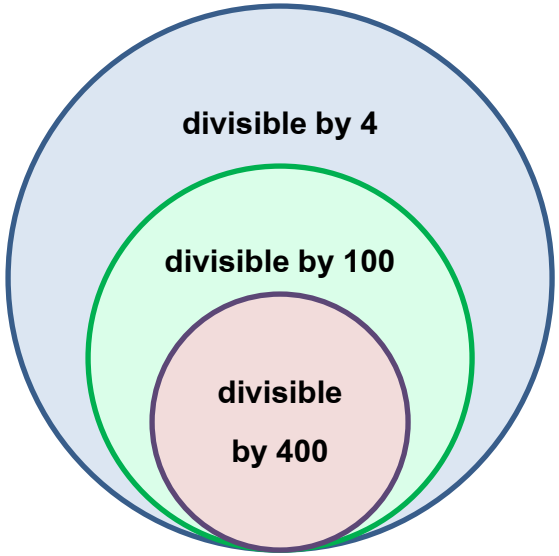
Example 1: Leap Year [7]

- STEP6: Reviewing

```
if (year % 4 == 0):
  if (year % 400 == 0):
    print("YES")
  elif (year % 100 == 0):
    print("NO")
  else:
    print("YES")

else:
  print("NO")
```

Where condition is a subset of each other (Not completely separated from each other.) Let's create a condition from a more specific case first. (small to large)



	Test Case	output
Case 1	2012 2016 2020	YES
Case 2	1700 1800 1900	NO
Case 3	1600 2000 2400	YES
Case 4	2013 2014 2015	NO