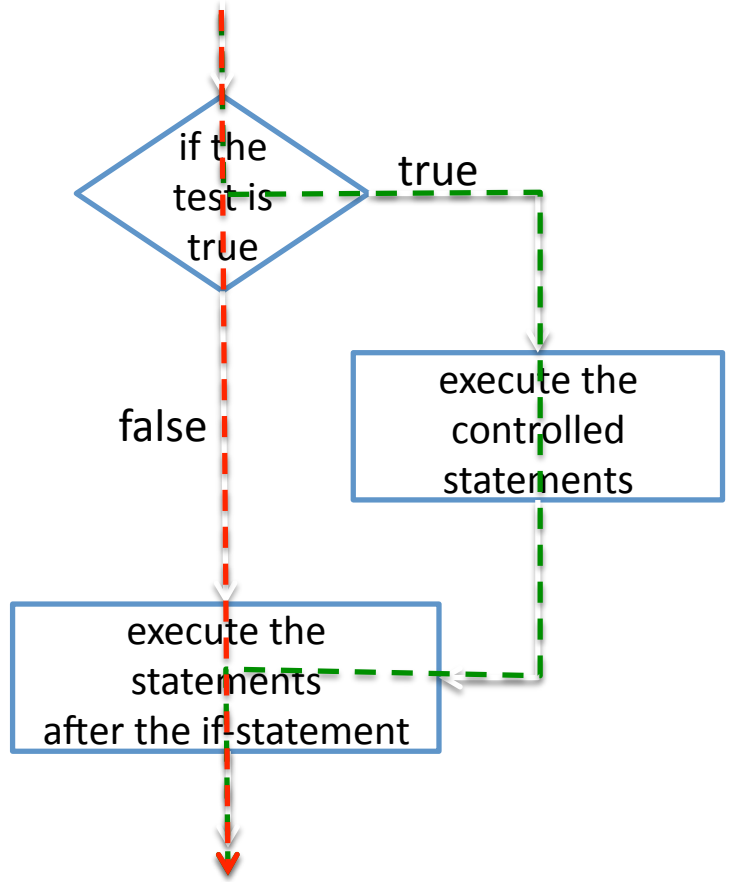
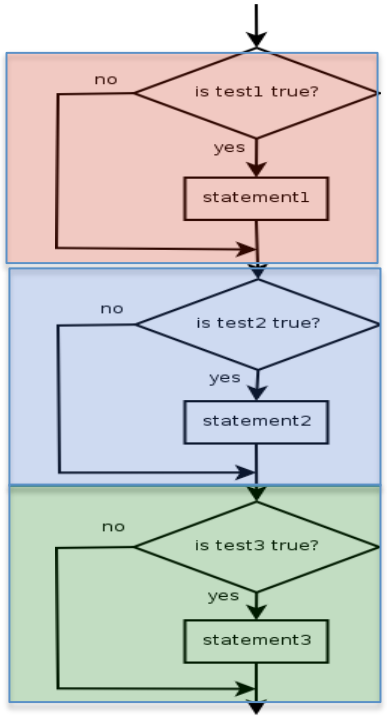
Basic if statement construct

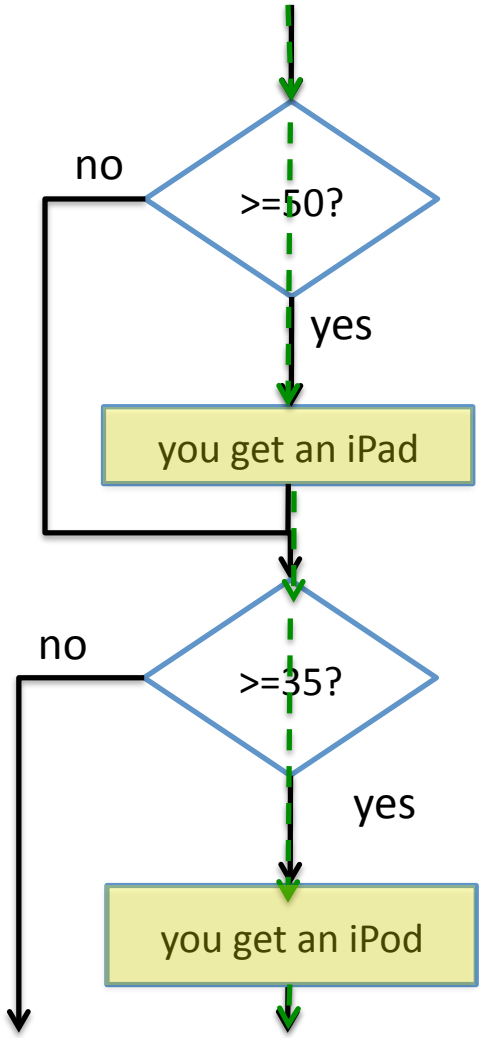


Can you spot the problem in this code?

```
Scanner console = new Scanner(System.in);
System.out.print(”How many magazines did you sell? ");
int percent = console.nextInt();
if (totalSales >= 50) {
    System.out.println(”Congratualtions, you win an iPad");
}
if (totalSales >= 35) {
    System.out.println("Congratualtions, you win an iPod");
}
if (totalSales >= 25) {
    System.out.println("Congratualtions, you win a stylus");
}
if (totalSales >= 15) {
    System.out.println("Congratualtions, you win a gift card");
} else {
    System.out.println(”Keep up the good work!");
}
...
```

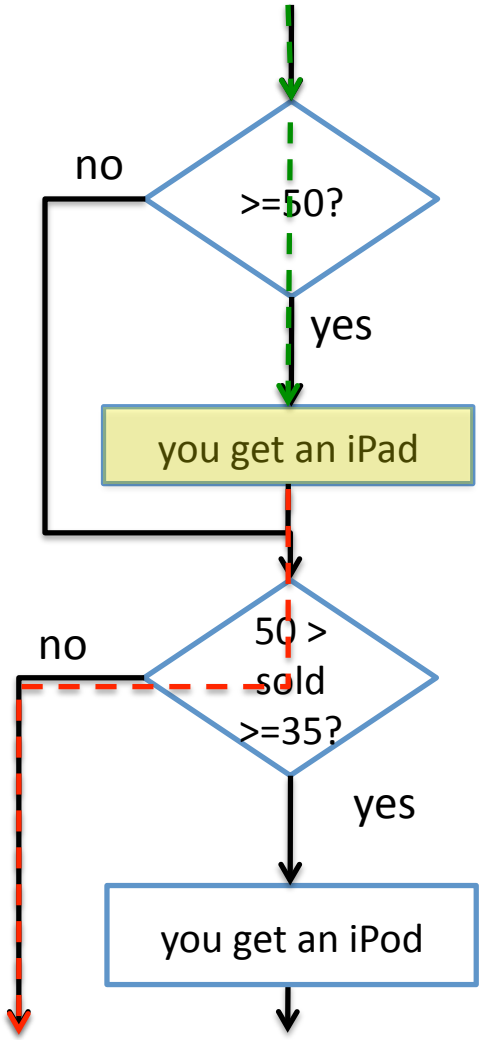


A closer look at the control flow



suppose you sold 72 magazines

Change the tests, making each one more explicit



What we really want…

```
Scanner console = new Scanner(System.in);
System.out.print(”How many magazines did you sell? ");
int percent = console.nextInt();
```

Greater than or equal to 50 - iPad

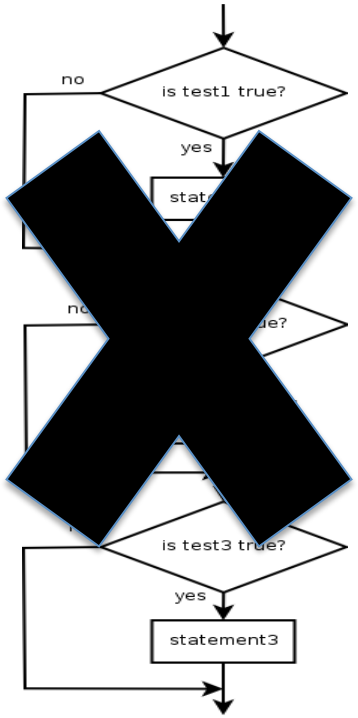
```
}
```

Otherwise, if greater than or equal to 35 - iPod

```
}
```

Otherwise, if greater than or equal to 25 - stylus

```
}
if (totalSales >= 15) {
    System.out.println("Congratualtions, you win a gift card");
} else {
    System.out.println(”Keep up the good work!");
}
...
```



Better, but poor style

```
    if (totalSales >= 50) {
        System.out.println(”Congratulations, you have won an iPod");
} else {
   if (totalSales >= 35) {
    System.out.println("Congratulations, you have won an iPod");
   } else {
     if (totalSales >= 25) {
       System.out.println("Congratulations, you have won a stylus");
     } else {
      if (totalSales >= 15) {
       System.out.println("Congratulations, you have won a gift card");
      } else {
         System.out.println(”Keep up the good work!");
      }
     }
   }
 }
```

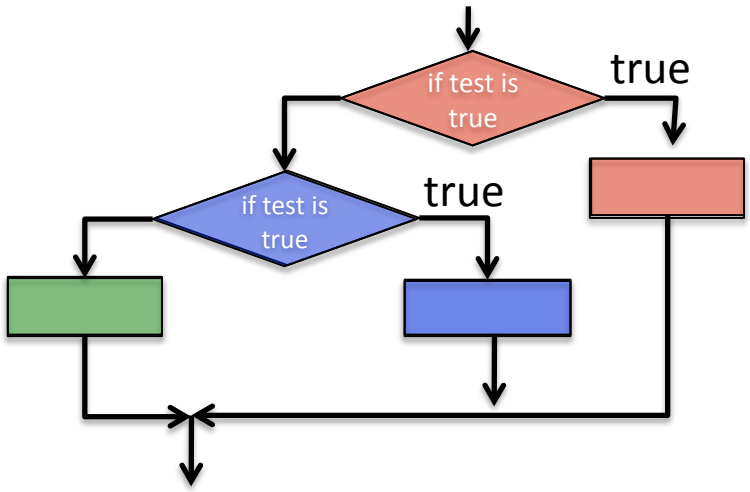
Try the multi-branch `if/else`

Instead of deeply nesting several tests, use else-if statements

Test the else-if when no other test has been true thus far

```
if (test) {
    statement(s);
} else if (test) {
    statement(s);
} else {
    statement(s);
}
```

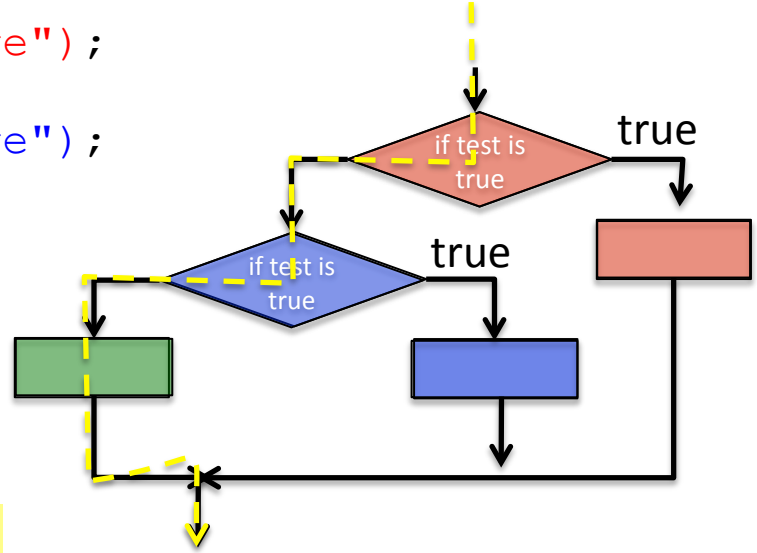
Here three statements can be controlled with two tests



Try the multi-branch `if/else`

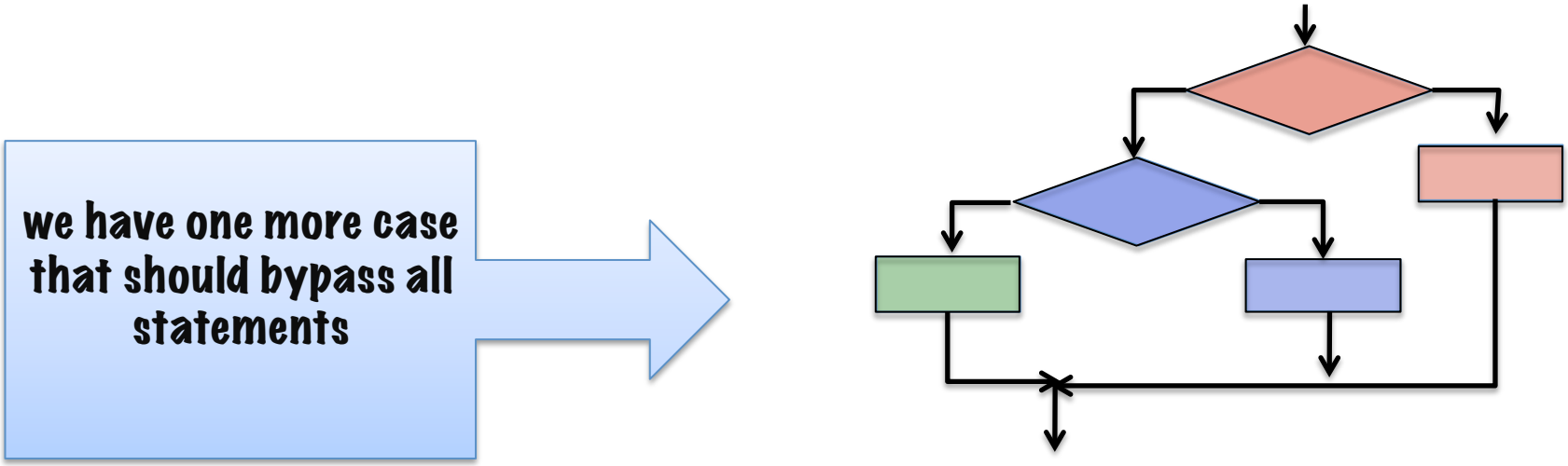
- Example:

```
if (inputX > 0) {
    System.out.println("Positive");
} else if (inputX < 0) {
    System.out.println("Negative");
} else {
    System.out.println("Zero");
}
```



There are three statements to execute, one for each condition. Two tests will suffice.

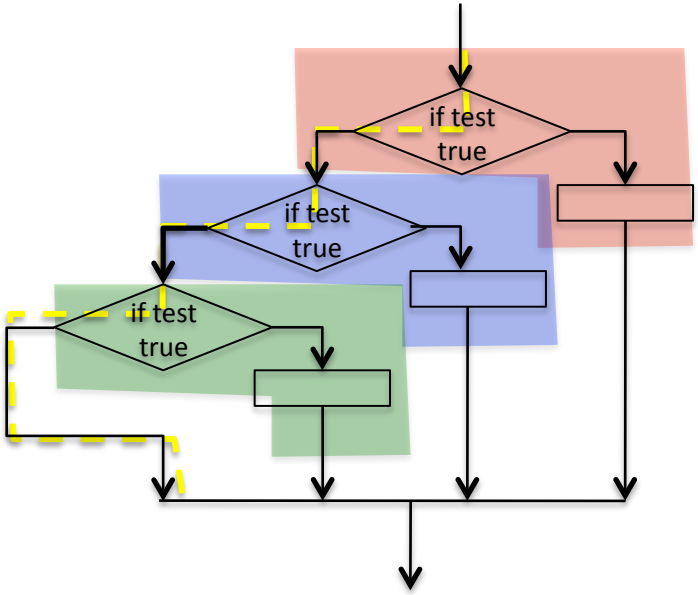
Suppose there are **three** statements, one for each condition and a **third case** where **no statement** should be executed. This model will not accommodate that situation



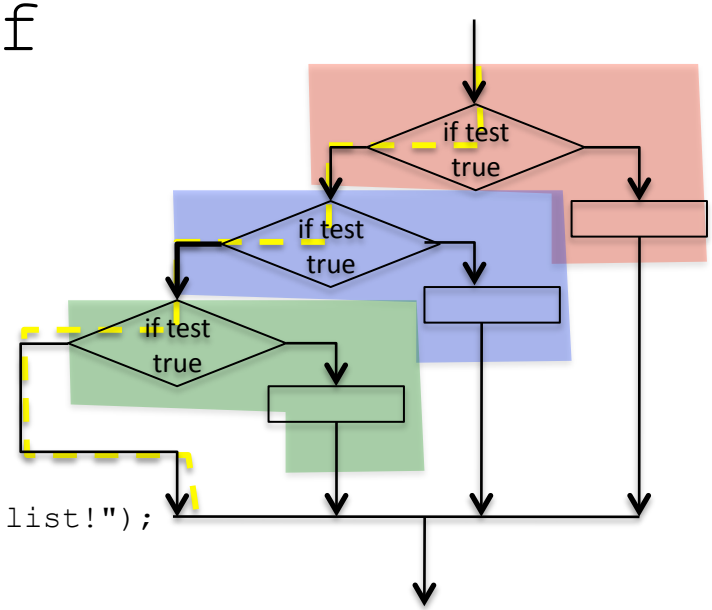
Multi-branch `if/else if`

- If the set of conditionals ends in `else`, one code path ***must*** be taken.
- If the set of conditionals ends with `if`, the program might not execute any path; there ***is*** a bypass.

```
if (test) {
    statement(s);
} else if (test) {
    statement(s);
} else if (test) {
    statement(s);
}
```



Multi-branch `if/else/if`

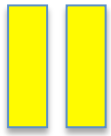


Example:

```
if (grade == 'A') {
    System.out.println("You are eligible for the Dean's list!");
} else if (grade == 'B') {
    System.out.println("You are eligible for the Honors list!");
} else if (grade == 'C') {
    System.out.println("You still have time to raise your
grade.");
}
```

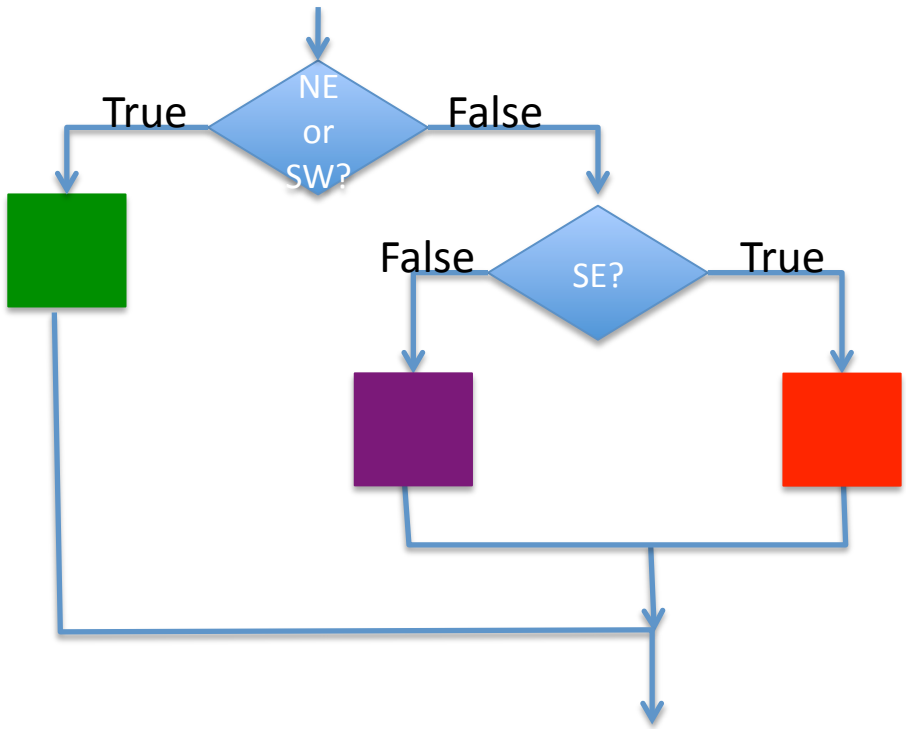
Review if statements

- Devise scenarios for different types of if statements
- Consider
  - How many different outcomes?
  - Should there be a path where no statement is executed?
  - Can you use well designed tests to combine conditions and eliminate if branches?
  - Can you reduce redundancy?



Returning to our game problem

- Three conditions: every player falls into one of these conditions
  - NW, purple
  - SE, red
  - SW, NE, green
- if/else if/else
- try writing this code



Assume a player's position is given as x,y coordiantes p1.x and p1.y

Note that

positive x * positive y = positive result, NE, green
negative x * negative y = positive result, SW, green
positive x * negative y = negative result, SE, red
negative x * positive y = negative result, NW, purple

positive x * positive y = positive result, NE, green
negative x * negative y = positive result, SW, green
positive x * negative y = negative result, SE, red
negative x * positive y = negative result, NW, purple

```
int position = p1.x * p1.y;
```

```
if (position >= 0) {
      p1.color = green;
} else if (p1.x > 0) {
      p1.color = red;
} else {
      p1.color = purple;
}
```

Only players with negative location will test the p1.x therefore the only two cases left are red or purple