## Jumps

<table>
<thead>
<tr>
<th>jump</th>
<th>[NUM:num]</th>
<th>Jump IP to position designated by num (typically a label)</th>
</tr>
</thead>
<tbody>
<tr>
<td>jump_if_0</td>
<td>[NUM:num1] [NUM:num2]</td>
<td>If num1 == 0, Jump IP to position designated by num2</td>
</tr>
<tr>
<td>jump_if_n0</td>
<td>[NUM:num1] [NUM:num2]</td>
<td>If num1 != 0, Jump IP to position designated by num2</td>
</tr>
</tbody>
</table>
Other Flow Control Structures - elif

```plaintext
val x = 0;
if (x < 0) print("-");
elif (x == 0) print("0");
else print("+");
```

```plaintext
val x = 0;
if (x < 0) print("-");
else if (x == 0) print("0");
else print("+");
```
Other Flow Control Structures - for

```plaintext
for (val x = 0; x > 5; x += 1) print(x);
```

```plaintext
val x = 0;
while (x > 5) {
    print(x);
    x += 1;
}
```
Other Flow Control Structures - switch

```scala
val x = 4;
switch (x - 1)
  case 1: x = 100;
  case 2: x = 200;
  default: x = 300;
```

```scala
val y = x - 1;
if (y == 1) x = 100;
else if (y == 2) x = 200;
else x = 300;
```
```
Which lines are wrong?

""Hi", I’m ";

array_set_size a1 11 #1
array_set_index a1 0 ' \"\' #2
array_set_index a1 1 'H'
array_set_index a1 2 'i'
array_set_index a1 3 '\"\' #2
array_set_index a1 4 ','
array_set_index a1 5 ','
array_set_index a1 6 'I'
array_set_index a1 7 '‘' #3
array_set_index a1 8 'm'
array_set_index a1 9 '
array_set_index a1 10 '‘’ #4
```
Implementing String Literals

1. Adjust tube.lex to capture string literals ("Say "Hello" 'Josh' !!! ")
2. In a ASTNode, figure out the char literals you need to output (one possible method):
   A. Convert lexeme string to a list of strings (each of length 1)
   B. Remove the bounding double quotes
   C. Make a new escaped vector, then for each character:
      1. If in “escaped-mode”, add a backslash and character to vector
         1. Unless it is a double-quote, then no back slash is needed
         2. if it is a backslash, go to “escaped-mode”
         3. If it is a character, add the character to vector
            1. If the character is a single-quote, add a backslash before it
Implementing String Literals (cont.)

1. Next create an array (of length of the escaped character vector) with array_set_size.

2. Then use array_set_index to fill string array with values (don’t forget about single-quotes around char literals).