Chapter 5: Conditional Statements
Decision Structures

• Some problems simply cannot be solved by performing a set of ordered steps, one after another (sequential execution)

• For example consider a company payroll program that determines whether an employee has worked overtime
  – If the employee has worked more than 40 hours, he or she gets paid a higher wage for the hours over 40
  – Otherwise, the overtime calculation should be skipped

• Solving this kind of problem requires a decision structure or conditional branching (a.ka. Selection statements)
Conditions

- Decision structures are based on a condition
- A condition is a logical (Boolean) expression that yields a value
- Conditions are typically written using the relational (comparison) operators.
- Boolean (logical) operators can also be used
Relational/Comparison Operators

>  Greater than
<  Less than
>= Greater than or Equal to
<= Less than or Equal to
== Equal to
!= Not Equal to

- Comparison operators pose a question and yield a value (true or false)
Relational/Comparison Operators

- True is stored in memory as 1
- False is stored in memory as 0
- Examples:
  - \(3 \geq 4\) false
  - \(5 > 0\) true
  - "A" != "a" true
  - \(5 + 3 = 8\) true
- **NOTE:** = is not the same as ==
  - Age = 19 #variable age stores value 19 (fact)
  - Age == 19 #checking if Age is equal to 19 (question)
### Boolean/Logical Operators

- **and**
- **or**
- **not**

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>P and Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>P or Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>P</th>
<th>not P</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
</tr>
</tbody>
</table>
Boolean/Logical Operators

- Examples:
  >>> P = (2 > 0)
  >>> Q = (10 <= 20)
  >>> print P
  >>> print Q
  >>> print (P and Q)
  >>> print (P or Q)
  >>> print (not P)
Conditions

- We use the relational operators and the Boolean operators to write conditions or questions, also known as Boolean expressions.

Example:
- age > 21
- today == “Tuesday” and time > 10

- Boolean expressions yield a value that is either true (1) or false (0)
Making decisions

- In a program we make decisions with the if statement:
  - **if** is a keyword
  - The keyword **if** is followed by a condition which ends with a colon:
  - Then follows a indented block of instructions
  - Example:
    ```
    if (red > 100) :
        red = red * 1.25
        blue = blue * 0.5
    ```
If statement

- Condition:
  - True
  - False

Instructions
How does the if statement work?

- First the condition is evaluated
- If the result of the condition is **true** the block of instructions is performed
- If the result of the condition is **false** the block of instructions is skipped (ignored)

**Conclusion:** the block of instructions associated to the **if** is performed **only** when the result of the condition is true
For loops and if statements

- We can always use an if statement inside a for loop (or vice versa)

```python
for p in getPixels(source):
    if (getRed(p) < getBlue(p)):
        setColor(p, newColor)
```
Distance between colors

• How do we measure distance between two points?
  • In the Cartesian coordinate system, the distance between two points is given by:
    \[ \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \]

• In JES the `distance()` function measures the distance between two colors:
  \[ \sqrt{(red_1 - red_2)^2 + (green_1 - green_2)^2 + (blue_1 - blue_2)^2} \]

Example: >>> dist = distance(color1, color2)
The `distance()` function

- JES provides us with a function that measures the “distance” between two colors
- This function receives two colors as the input and it returns a number
- Example:
  ```python
  >>> d = distance (blue, red)
  >>> print d
  >>> 360.624
  ```
Threshold values

- Sometimes we need to know if two values are “pretty close” so we can consider them to be “equal”
- In these cases a good rule is to find out if the values are “close enough” by using a threshold value.
- Example:

```python
if (distance (red, myColor) < 165):
```
- Here the threshold value is 165. If the distance between red and myColor is less than 165 we will consider both colors to be “close enough”
- You as a programmer decide what is a good threshold value
  - It may take some experimentation to find a good value
Example

removeRedEye(pic, startX, startY, endX, endY, newColor)

- The use of parameters in this function, makes it a very general, re-usable function.
if..... else: choosing between two set of instructions
if... else

- format:
  ```python
  if some_condition :
      Instructions
  else :
      Other instructions
  ```

- Example:
  ```python
  if (red > 100) :
      red = red * 1.25
  else
      red = red * 0.25
  ```

- Example: `posterizeGrey (picture)`

**NOTE:** posterizing ➔ reducing the number of colors in the picture
Choosing among multiple set of instructions

- **condition1**
  - False: Other Instructions 2
  - True: Instructions

- **condition2**
  - False: Other Instructions 1
  - True: Instructions
**elif**

- **format:**
  ```python
  if condition1 :
    Instructions
  elif condition2 :
    Other instructions1
  else :
    Other instructions2
  ```

- **Example:**
  ```python
  if (red > 50 and red < 100) :
    red = red * 1.25
  elif (red >= 100 and red < 200) :
    red = red * 0.25
  else :
    red = red * 0.10
  ```

- **Example:** `posterize(picture)`
Nested if

- As with for loops, if statements can be nested.

- Example:

  ```python
  if (red > 191):
      red = red * 1.08
  if (red > 255):  # cap red channel to 255, the max
      red = 255
  blue = blue * 0.9
  ```

- How does it work? Evaluate the outer if statement first, if true then work on it’s block. When an inner if statement is found inside the outer block, evaluate inner if statement, if true execute it’s block, otherwise skip it.

- Example: `sepiaTint(picture)`
Expressions inside a condition

- A condition can contain an expression.
- The final result of the condition is always true or false.
- Example:
  
  ```java
  if (x + y + z) > 30 :
  ```

- Chromakey example: background color of a picture is replaced with another background, while the foreground of the original pictures stays.
  - It’s easier to do with an original background that is green
    - there’s less overlap with common colors
  - Pictures must be of the same size
Some new JES commands and keywords

- Predefined colors: black, white, blue, red, green, gray, lightGray, darkGray, yellow, orange, pink, magenta, cyan.

- Drawing functions:
  
  \begin{itemize}
  \item \texttt{addText(pict, x, y, string, color)}
  \item \texttt{addLine(pict, x1, y1, x2, y2, color)}
  \item \texttt{addRect(pict, x, y, width, height, color)}
  \item \texttt{addRectFilled(pict, x, y, width, height, color)}
  \item \texttt{addArc(pict, x, y, width, height, start, angle, color)}
  \item \texttt{addArcFilled(pict, x, y, width, height, start, angle, color)}
  \item \texttt{addOval(pict, x, y, width, height, color)}
  \item \texttt{addOvalFilled(pict, x, y, width, height, color)}
  \end{itemize}

- Drawing example
One more thing about range and for loops

- Negative increase = decrease
  >>> print range (25, 0, -1)

- Increasing/decreasing by more than just 1
  >>> print range (0, 25, 2)
  >>> print range (25, 0, -2)

- Combine these ideas with a for loop:
  for index in range (25, 0, -2)

- Example: coolPic()
Example: Edge detection

- We are going to compare each pixel luminance to the pixel below it and to the right of it.

<table>
<thead>
<tr>
<th>here</th>
<th>right</th>
</tr>
</thead>
<tbody>
<tr>
<td>x, y</td>
<td>x+1, y</td>
</tr>
<tr>
<td>down</td>
<td></td>
</tr>
<tr>
<td>x, y+1</td>
<td></td>
</tr>
</tbody>
</table>

- If there is a *suitable difference* in luminance below and to the right, we will make the pixel black, otherwise, we will make it white.

- Here again we will use a threshold value, this time to check if the difference between two values is “close enough”
Debugging your programs

- A bug is an error in your program
- There are two kinds of errors
  - Syntax errors
  - Logic or semantic errors
Syntax Errors

- A syntax error is an error that occurs when a program cannot understand a command that has been entered.
- This happens when a statement in the program violates the rules of the programming language
- Examples:
  ```python
  def myFuntion(pic)  #colon missing at the end of the line
  For x in range (0, getWidth(pic)): #For is not a keyword
  ```
- A syntax error must be fixed before the program can be executed
Logic / Semantic Errors

- A logic or semantic error causes the program to operate incorrectly, but not to fail
  - that is, you can still run the program but you will get erroneous or unintended results.

- Example:
  
  ```python
  if (red < 100 and red > 200) :  #probably meant to use or
  ```

- Since the program will still run, the programmer must be careful examining the results of the program to detect if there is a mistake in the logic of it
  - Lucky for us, this kind of mistakes should be easy to spot in our images resulting from our functions
Midterm Exam

- **Date:** Tuesday October 28
- **Time:** During the lecture period
- **Room:** Curtis 342
- Exam is closed book/ closed notes
  - A handout with the Python and JES commands will be provided
- **You need to bring:**
  - Pencil, eraser, and student ID
- **Study Guide is posted in Learn**