CS A131: Lecture 4

Nadia Ahmed

Orange Coast College, Computer Science

CS A131

Overview

- Relational Operators
- Logical Operators
- Ternary Operator Expressions
- Conditional Operators
- Selection
  - `if`
  - `if-else`
  - `if-elif-else`

Comparison of Values

- Relational Operators
  - direct comparison of two values
  - Boolean result: truth value, True or False
- Logical Operators
  - Operations on Boolean values
- Conditional Operators
  - Conditional evaluation of expressions

Comparison operations

- Comparison operations
  - `<` less than
  - `>` greater than
  - `<=` less than or equal to
  - `>=` greater than or equal to
  - `==` equal to (remember = means assignment!)
  - `!=` not equal to
- Comparison is defined for all basic types
  - integer (e.g. `5 < 6`)
  - floating point (e.g. `7.0 < 7e1`)
- Result type is Boolean, but represented as integer
  - `False` `0`
  - `True` `1` (or any other value *not* equal to zero)

Logical Operators

- Operation on Boolean/truth values
  - `not` logical negation
  - `and` logical and
  - `or` logical or
- Truth table

`x`	`y`	`not x`	`x and y`	`x or y`
`False`	`False`	`True`	`False`	`False`
`False`	`True`	`True`	`False`	`True`
`True`	`False`	`False`	`False`	`True`
`True`	`True`	`False`	`True`	`True`

- Argument and result types are Boolean but can be represented as integer
  - `False` 0
  - `True` 1 (or any other value *not* equal to zero)

`bool` Data Type and Logical (Boolean) Expressions

- The data type `bool` has logical (Boolean) values `True` and `False`
- `bool`, `True`, `False` are reserved words.
- The identifier `True` has the value `1`
- The identifier `False` has the value `0`

Ternary Operator Expressions

- Conditional evaluation of values in expressions `true-value if test else false-value`
  - evaluates the `test`
  - if `test` is true, then the result is `true-value`
  - otherwise, the result is `false-value`
- Examples
  - `42 if (4<5) else (4 + 8)` evaluates to 42
  - `x if (2 == 1 + 2) else (y)` evaluates to `y`
  - `-x if x < 0 else x` evaluates to `abs(x)`

Operator Evaluation Order

- Associativity: left to right or right to left
- Precedence: group-wise, top to bottom

parentheses	`(, )`	n/a
unary plus, minus, negation	`+, -, not`	right to left
type casting	`typename()`	right to left
multiplication, division, modulo	`*, /, %`	left to right
addition, subtraction	`+, -`	left to right
shift left, shift right	`<<, >>`	left to right
relational operators	`<, <=, >=, >`	left to right
equality	`==, !=`	left to right
logical and	`and`	left to right
logical or	`or`	left to right
assignment operator	`=`	right to left

Conditional Statements

`if` statement

- Control flow statement for decision making
  - Changes control flow depending on a specified condition
- Example:

```
if x < 0:
  print("%f is negative" % x)
if x >= 0:
  print("%f  is positive" % x)
```

- Syntax: `if` construct consists of
  - Keyword `if`
  - Condition expression evaluated to true or false
  - Body statement block
- Semantics
  - Body is executed *only if* the condition evaluates to `True`

comparison.py

```
######################################
# comparison.py: arithmetic comparisons
#
# author: Nadia Ahmed
#
# modifications:
# 07/06/2016 NA initial version
######################################

def main():
  # input section
  a = int(input("Please enter a value for integer a: "))
  b = int(input("Please enter a value for integer b: "))
  ...
```

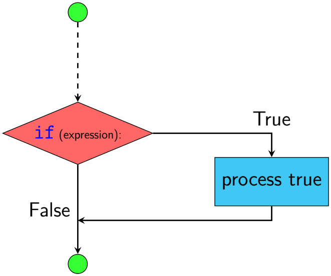
comparison.py

```
...
  # computation section
  if a == b:
    print(" %d is equal to %d" % (a,b))
  #fi
  if a != b:
    print("%d is not equal to %d" % (a,b))
  #fi
  if a < b:
    print("%d  is less than %d" % (a,b))
  #fi
  if a > b:
    print("%d is greater than %d" % (a,b))
  #fi
  if a <=b:
    print( "%d is less than or equal to %d" % (a,b))
  #fi
  if a >=b:
    print("%d is greater than or equal to %d" % (a,b))
  #fi
# EOF

#call main
main()
```

Selection: if statement

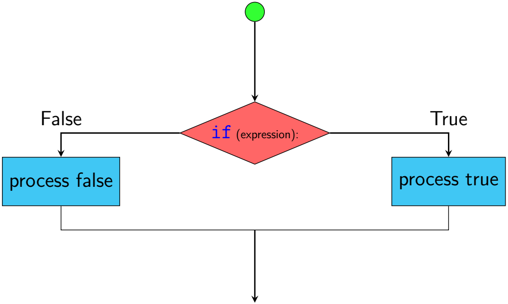
- Flow chart:



- Example:

```
if grade >= 60:
  print("You passed.")
# fi
```

Selection: if-else statement



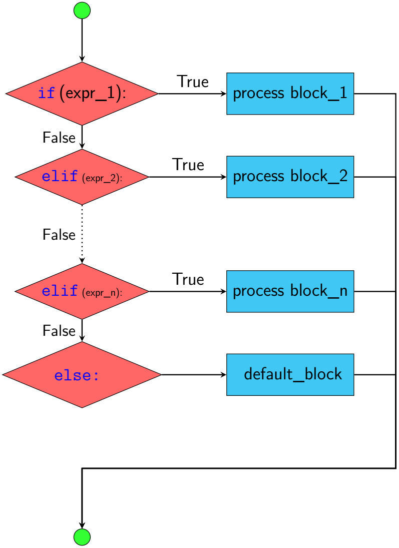
- Flow chart:
- Example:

```
if grade >= 60:
  print("You passed.")
# fi
else:
  print("You failed.")
# esle
```

- Make sure the if clause and the else clause are aligned.
- Statements within clauses should be indented.

Multiple Selections: `if elif else` statement

- Flow chart:



- Example:

```
if grade >= 90:
  print("Excellent!")
# fi
elif grade >= 80:
  print("Satisfactory.")
  .
  .
  .
# file
elif grade >= 60:
  print("You passed.")
# file
else:
  print("Failed.")
# esle
```

Comparing Strings

- You can compare two strings to see if they are equal using ==
- You can determine whether one string is greater than > or less than < another which is useful for sorting.
- ASCII codes allow us to represent strings
    - The uppercase characters A through Z are represented by the numbers 65 through 90.
    - The lowercase characters a through z are represented by the numbers 97 through 122.
    - When the digits 0 through 9 are stored in memory as characters, they are represented by the numbers 48 through 57. (For example, the string 'abc123' would be stored in memory as the codes 97, 98, 99, 49, 50, and 51.)
    - A blank space is represented by the number 32.