Outline

1. Boolean Algebra
   - logical expressions
   - pseudocode and flowcharts
2. Conditional Constructs
   - the type `bool`
   - if, else, elif

MCS 260 Lecture 8
Introduction to Computer Science
Jan Verschelde, 21 June 2023

Boolean Algebra, Flowcharts Conditional Expressions

1. Boolean Algebra
   - logical expressions
   - pseudocode and flowcharts
2. Conditional Constructs
   - the type `bool`
   - if, else, elif

Boolean Algebra

computing with logical expressions

Boolean algebra is the calculation with `True` and `False` (often having values 1 and 0). The operators are `and`, `or`, and `not`. Truth tables define the outcome for all values:

`x`	`y`	`x and y`
`False`	`False`	`False`
`False`	`True`	`False`
`True`	`False`	`False`
`True`	`True`	`True`

`x`	`y`	`x or y`
`False`	`False`	`False`
`False`	`True`	`True`
`True`	`False`	`True`
`True`	`True`	`True`

Evaluation Laws

law and order in the Boolean algebra

When `not`, `and`, `or` occur in an expression,
`not` is first evaluated, before `and`, and finally `or`.

De Morgan's laws for simplifying expressions:

- `not (( not x ) or ( not y ))` = `x and y`
  Negating not being alive or not being well
  means being alive and being well.
- `not (( not x ) and ( not y ))` = `x or y`
  Negating not going to school and not going to work
  means going to school or going to work.

We prove these laws by truth tables.
Applications of truth tables:

- Realization of electronic circuits.
- Simplification of conditional statements in programs.

Boolean Algebra, Flowcharts
Conditional Expressions

1. Boolean Algebra
   - logical expressions
   - pseudocode and flowcharts

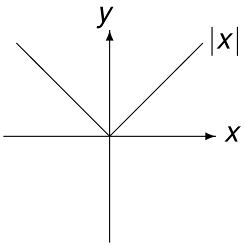
2. Conditional Constructs
   - the type `bool`
   - if, else, elif

The Absolute Value – an example of an `if` statement

The function `abs` is available in Python:

```
>>> abs(-3.5)
3.5
>>> abs(3.5)
3.5
```

The mathematical definition of $\texttt{abs}(x)$ as $y = |x|$:



$$|x| = \begin{cases} x & \text{if } x \geq 0 \\ -x & \text{if } x < 0 \end{cases}$$

Pseudocode

to formally describe algorithms

To develop and define an algorithm, we use *pseudocode*.
Pseudocode is not real code, but to the reader it has the same properties as a formal language.

Example: print the absolute value of a number.
The number is given by the user.

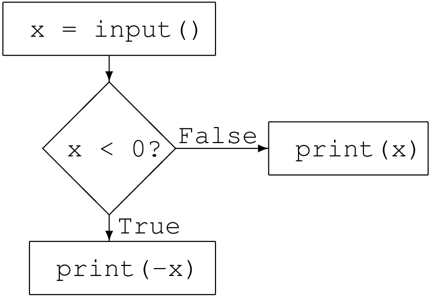
In words, we could describe the program as:

```
ask the user for a number
if the number is less than zero
 then print –
      print the number
 else print the number
```

Flowcharts

pictures of algorithms

Printing the absolute value of a number:



Flowcharts schematically represent the logical flow.

Boolean Algebra, Flowcharts
Conditional Expressions

1. Boolean Algebra
   - logical expressions
   - pseudocode and flowcharts

2. Conditional Constructs
   - the type `bool`
   - if, else, elif

logical expressions

- Variables of the type `bool` are either `True` or `False`, with corresponding values 1 or 0.
- Observe the differences between
  - comparison `==` and assignment `=`
  - comparison `==` of values and `is` for objects
- A dictionary is the counterpart to the `if else` statement:
  - keys in the dictionary are `True` and `False`,
  - values are the strings which store the code in `if else` cases.

Boolean Algebra, Flowcharts Conditional Expressions

1. Boolean Algebra
   - logical expressions
   - pseudocode and flowcharts
2. Conditional Constructs
   - the type `bool`
   - if, else, elif

The if Statement – conditional execution of code

The syntax of the `if`:

```
if < condition >:
    < statements when condition is True >
```

All statements to be executed only if the condition is true must be preceded *by the right intendations!*

Suppose we want to print the '+' for positive numbers.

With an `if` we could do it as follows:

```
if x > 0:
    print('+')
    print(x)
```

```
if x > 0:
    print('+')
print(x)
```

Only the second one works correctly for all `x`.

the if else statement

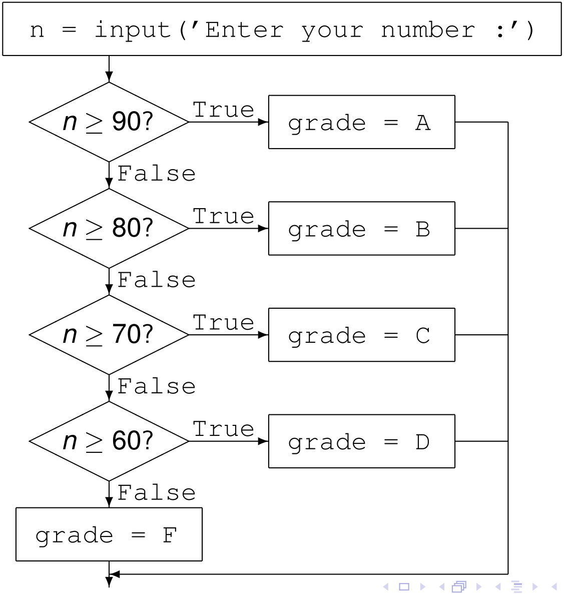
The syntax of the `if else`:

```
if < condition >:
    < statements when condition is True >
else:
    < statements when condition is False >
```

Printing the absolute value of a number:

```
if x < 0:
    print(-x)
else:
    print(x)
```

flowchart of grade scale



`if elif else` for multiple alternatives

The syntax of the `if elif else`:

```
if < condition 1 >:
    < statements when condition 1 is True >
elif < condition 2 >:
    < statements when condition 2 is True >
...
elif < condition n >:
    < statements when condition n is True >
else:
    < statements when everything is False >
```

The conditions are evaluated in the order as they appear.

Exercises

1. Omit the brackets in De Morgan's Laws
   and create a truth table to evaluate the expressions.
2. Draw a flowchart for the code using a nested if else for the followup questions `"happy ?"` and `"bored ?"`.
3. Write pseudocode and draw of flowchart for a program that reads in a positive number and prints out whether the number is divisible by 2, 3, 5, or not.
4. Define a Python dictionary for the truth table of `x and y`.