

Computer Architecture

- 1 Computer Architecture
  - Hardware Components
  - Programming Environments

Hardware Components  
Programming Environments

Getting Started with Python

- 2 Getting Started with Python
  - Installing Python
  - Executing Python code

Installing Python  
Executing Python code

Number Systems

- 3 Number Systems
  - Decimal and Binary notations

Decimal and Binary notations

Running Sage

- 4 Running Sage

Summary + Assignments

- 5 Summary + Assignments

MCS 260 Lecture 2  
Introduction to Computer Science  
Jan Verschelde, 13 January 2010

# Computer Architecture

## Hardware & Software

### Computer Architecture

Hardware  
Components

Programming  
Environments

### Getting Started with Python

Installing Python  
Executing Python code

### Number Systems

Decimal and Binary notations

### Running Sage

### Summary + Assignments

A computer system consists of

- 1 **Hardware: physical components of computer**
  - computer: processor, memory, bus, ...
  - peripherals: printer, screen, keyboard, mouse, ...
- 2 **Software: programs executed by computer**
  - basic software like the Operating System (OS) either Unix (e.g.: Solaris, GNU-Linux, Mac OS X) or Windows (the OS of Microsoft)
  - application software such as IDLE, Sage, ...  
application software needs operating system to run

# Computer Architecture

## Hardware & Software

### Computer Architecture

Hardware  
Components

Programming  
Environments

### Getting Started with Python

Installing Python  
Executing Python code

### Number Systems

Decimal and Binary notations

### Running Sage

### Summary + Assignments

A computer system consists of

- 1 Hardware: physical components of computer
  - computer: processor, memory, bus, ...
  - peripherals: printer, screen, keyboard, mouse, ...
- 2 Software: programs executed by computer
  - basic software like the Operating System (OS) either Unix (e.g.: Solaris, GNU-Linux, Mac OS X) or Windows (the OS of Microsoft)
  - application software such as IDLE, Sage, ...  
application software needs operating system to run

# Computer Architecture first steps with Python

Computer  
Architecture

Hardware  
Components

Programming  
Environments

Getting  
Started with  
Python

Installing Python

Executing Python  
code

Number  
Systems

Decimal and Binary  
notations

Running Sage

Summary +  
Assignments

- 1 Computer Architecture  
Hardware Components  
Programming Environments
- 2 Getting Started with Python  
Installing Python  
Executing Python code
- 3 Number Systems  
Decimal and Binary notations
- 4 Running Sage
- 5 Summary + Assignments

# Hardware Components

## Computer Architecture

### Hardware Components

Programming Environments

## Getting Started with Python

Installing Python  
Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

**processor** (or CPU: Central Processing Unit) does the computing and coordinates data transfer

**memory** (or RAM: Random Access Memory) is used to store data and programs, of limited capacity and volatile (lost if power off)

**storage** persistently stores large quantities of data and programs, slower access to storage than to memory, but larger than RAM

**peripherals** are used to communicate with computer

**system bus** connects CPU, RAM, storage, and peripherals

# Hardware Components

## Computer Architecture

### Hardware Components

Programming Environments

## Getting Started with Python

Installing Python  
Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

**processor** (or CPU: Central Processing Unit) does the computing and coordinates data transfer

**memory** (or RAM: Random Access Memory) is used to store data and programs, of limited capacity and volatile (lost if power off)

**storage** persistently stores large quantities of data and programs, slower access to storage than to memory, but larger than RAM

**peripherals** are used to communicate with computer

**system bus** connects CPU, RAM, storage, and peripherals

# Hardware Components

## Computer Architecture

### Hardware Components

Programming Environments

## Getting Started with Python

Installing Python  
Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

**processor** (or CPU: Central Processing Unit) does the computing and coordinates data transfer

**memory** (or RAM: Random Access Memory) is used to store data and programs, of limited capacity and volatile (lost if power off)

**storage** persistently stores large quantities of data and programs, slower access to storage than to memory, but larger than RAM

**peripherals** are used to communicate with computer

**system bus** connects CPU, RAM, storage, and peripherals

# Hardware Components

## Computer Architecture

### Hardware Components

Programming Environments

## Getting Started with Python

Installing Python  
Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

**processor** (or CPU: Central Processing Unit) does the computing and coordinates data transfer

**memory** (or RAM: Random Access Memory) is used to store data and programs, of limited capacity and volatile (lost if power off)

**storage** persistently stores large quantities of data and programs, slower access to storage than to memory, but larger than RAM

**peripherals** are used to communicate with computer

**system bus** connects CPU, RAM, storage, and peripherals

# Hardware Components

## Computer Architecture

### Hardware Components

#### Programming Environments

## Getting Started with Python

### Installing Python Executing Python code

## Number Systems

### Decimal and Binary notations

## Running Sage

## Summary + Assignments

- processor** (or CPU: Central Processing Unit) does the computing and coordinates data transfer
- memory** (or RAM: Random Access Memory) is used to store data and programs, of limited capacity and volatile (lost if power off)
- storage** persistently stores large quantities of data and programs, slower access to storage than to memory, but larger than RAM
- peripherals** are used to communicate with computer
- system bus** connects CPU, RAM, storage, and peripherals

# Computer Architecture first steps with Python

Computer  
Architecture

Hardware  
Components

Programming  
Environments

Getting  
Started with  
Python

Installing Python

Executing Python  
code

Number  
Systems

Decimal and Binary  
notations

Running Sage

Summary +  
Assignments

- 1 Computer Architecture
  - Hardware Components
  - Programming Environments
- 2 Getting Started with Python
  - Installing Python
  - Executing Python code
- 3 Number Systems
  - Decimal and Binary notations
- 4 Running Sage
- 5 Summary + Assignments

# Programming Environments

what it takes to run programs

**editor** : is used to write source code

**compiler** : translates source code into an object, an executable program — if code is bug free

**interpreter** : executes high level code directly

**linker** : combines several objects into one single executable program

**debugger** : helps user to locate bugs, allowing a stepwise execution of the program

use an IDE: Integrated Development Environment  
Python's IDE is called IDLE

# Programming Environments

what it takes to run programs

**editor** : is used to write source code

**compiler** : translates source code into an object, an executable program — if code is bug free

**interpreter** : executes high level code directly

**linker** : combines several objects into one single executable program

**debugger** : helps user to locate bugs, allowing a stepwise execution of the program

use an IDE: Integrated Development Environment  
Python's IDE is called IDLE

# Programming Environments

what it takes to run programs

**editor** : is used to write source code

**compiler** : translates source code into an object, an executable program — if code is bug free

**interpreter** : executes high level code directly

**linker** : combines several objects into one single executable program

**debugger** : helps user to locate bugs, allowing a stepwise execution of the program

use an IDE: Integrated Development Environment  
Python's IDE is called IDLE

# Programming Environments

what it takes to run programs

**editor** : is used to write source code

**compiler** : translates source code into an object, an executable program — if code is bug free

**interpreter** : executes high level code directly

**linker** : combines several objects into one single executable program

**debugger** : helps user to locate bugs, allowing a stepwise execution of the program

use an IDE: Integrated Development Environment  
Python's IDE is called IDLE

# Programming Environments

what it takes to run programs

**editor** : is used to write source code

**compiler** : translates source code into an object, an executable program — if code is bug free

**interpreter** : executes high level code directly

**linker** : combines several objects into one single executable program

**debugger** : helps user to locate bugs, allowing a stepwise execution of the program

use an IDE: Integrated Development Environment  
Python's IDE is called IDLE

# Programming Environments

what it takes to run programs

**editor** : is used to write source code

**compiler** : translates source code into an object, an executable program — if code is bug free

**interpreter** : executes high level code directly

**linker** : combines several objects into one single executable program

**debugger** : helps user to locate bugs, allowing a stepwise execution of the program

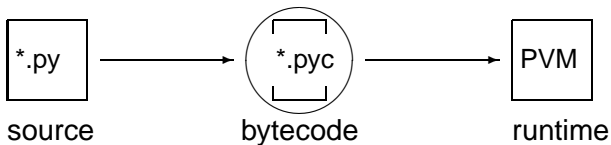
use an IDE: Integrated Development Environment  
Python's IDE is called IDLE

# Executing Programs

how programs are executed

- high level programming languages are oriented towards the convenience of the programmer
- an assembler language offers symbols to the basic instructions for writing machine code

The Python Virtual Machine:



The Python interpreter creates bytecode that is then executed by the Python Virtual Machine at runtime.

# Computer Architecture first steps with Python

Computer  
Architecture

Hardware  
Components

Programming  
Environments

Getting  
Started with  
Python

Installing Python

Executing Python  
code

Number  
Systems

Decimal and Binary  
notations

Running Sage

Summary +  
Assignments

- 1 Computer Architecture
  - Hardware Components
  - Programming Environments
- 2 Getting Started with Python
  - Installing Python
  - Executing Python code
- 3 Number Systems
  - Decimal and Binary notations
- 4 Running Sage
- 5 Summary + Assignments

# Installing Python

on your desktop or laptop

Free to download from [www.python.org](http://www.python.org).

**Unix** most Linux distributions have Python installed, or else contact your system administrator. Login to [icarus.cc.uic.edu](http://icarus.cc.uic.edu) using your netid.

**Mac OS X** like with unix you can download the source or run Python 2.6.4 for Macintosh OS X (universal installer both for PPC and IntelMacs). Computers in ACCC labs (should) have Python installed.

**Windows** run the Python 2.6.4 windows installer. Most labs on campus have Python installed.

# Computer Architecture first steps with Python

## Computer Architecture

Hardware  
Components

Programming  
Environments

## Getting Started with Python

Installing Python

**Executing Python  
code**

## Number Systems

Decimal and Binary  
notations

## Running Sage

## Summary + Assignments

- 1 Computer Architecture
  - Hardware Components
  - Programming Environments
- 2 Getting Started with Python
  - Installing Python
  - Executing Python code**
- 3 Number Systems
  - Decimal and Binary notations
- 4 Running Sage
- 5 Summary + Assignments

# Executing Python code

program prints "hello world!"

## Computer Architecture

Hardware Components

Programming Environments

## Getting Started with Python

Installing Python

**Executing Python code**

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

### Ways to run python programs:

- 1 In a Python session, type commands at the prompt:  

```
>>> print 'hello world!'
```
- 2 Running programs at the command prompt:
  - 1 Save Python commands in a file, e.g.: `hello.py`.
  - 2 Type `python hello.py` at the command prompt.
- 3 On windows, double click a file with `.py` extension.
- 4 In IDLE, go to the `run` menu in the editor.

# Executing Python code

program prints "hello world!"

Computer  
Architecture

Hardware  
Components

Programming  
Environments

Getting  
Started with  
Python

Installing Python

Executing Python  
code

Number  
Systems

Decimal and Binary  
notations

Running Sage

Summary +  
Assignments

## Ways to run python programs:

- 1 In a Python session, type commands at the prompt:

```
>>> print 'hello world!'
```

- 2 Running programs at the command prompt:

- 1 Save Python commands in a file, e.g.: `hello.py`.

- 2 Type `python hello.py` at the command prompt.

- 3 On windows, double click a file with `.py` extension.

- 4 In IDLE, go to the `run` menu in the editor.

# Executing Python code

program prints "hello world!"

Computer  
Architecture

Hardware  
Components

Programming  
Environments

Getting  
Started with  
Python

Installing Python

Executing Python  
code

Number  
Systems

Decimal and Binary  
notations

Running Sage

Summary +  
Assignments

## Ways to run python programs:

- 1 In a Python session, type commands at the prompt:

```
>>> print 'hello world!'
```

- 2 Running programs at the command prompt:

- 1 Save Python commands in a file, e.g.: `hello.py`.

- 2 Type `python hello.py` at the command prompt.

- 3 On windows, double click a file with `.py` extension.

- 4 In IDLE, go to the `run` menu in the editor.

# Executing Python code

program prints "hello world!"

Computer  
Architecture

Hardware  
Components

Programming  
Environments

Getting  
Started with  
Python

Installing Python

Executing Python  
code

Number  
Systems

Decimal and Binary  
notations

Running Sage

Summary +  
Assignments

## Ways to run python programs:

- 1 In a Python session, type commands at the prompt:

```
>>> print 'hello world!'
```

- 2 Running programs at the command prompt:

- 1 Save Python commands in a file, e.g.: `hello.py`.

- 2 Type `python hello.py` at the command prompt.

- 3 On windows, double click a file with `.py` extension.

- 4 In IDLE, go to the `run` menu in the editor.

# Executing Python code

program prints "hello world!"

Computer  
Architecture

Hardware  
Components

Programming  
Environments

Getting  
Started with  
Python

Installing Python

Executing Python  
code

Number  
Systems

Decimal and Binary  
notations

Running Sage

Summary +  
Assignments

## Ways to run python programs:

- 1 In a Python session, type commands at the prompt:

```
>>> print 'hello world!'
```

- 2 Running programs at the command prompt:

- 1 Save Python commands in a file, e.g.: `hello.py`.

- 2 Type `python hello.py` at the command prompt.

- 3 On windows, double click a file with `.py` extension.

- 4 In IDLE, go to the `run` menu in the editor.

# Executing Python code

program prints "hello world!"

Computer  
Architecture

Hardware  
Components

Programming  
Environments

Getting  
Started with  
Python

Installing Python

Executing Python  
code

Number  
Systems

Decimal and Binary  
notations

Running Sage

Summary +  
Assignments

## Ways to run python programs:

- 1 In a Python session, type commands at the prompt:  

```
>>> print 'hello world!'
```
- 2 Running programs at the command prompt:
  - 1 Save Python commands in a file, e.g.: `hello.py`.
  - 2 Type `python hello.py` at the command prompt.
- 3 On windows, double click a file with `.py` extension.
- 4 In IDLE, go to the `run` menu in the editor.

# Interactive Python code

Let us write a program that asks for our name, as input.  
And then, as output, writes hello followed by our name.

**input** The `raw_input()` function accepts only text input:

```
>>> name = raw_input('Who\'s there ? ')
```

Displays `Who's there ?` on screen and assigns what the user types in to the variable `name`.

**output** With `name` in its argument, the `print` command displays the value of `name`:

```
>>> print 'hello ', name , '!'
```

*develop Python code interactively at the prompt*

# Interactive Python code

Let us write a program that asks for our name, as input.  
And then, as output, writes hello followed by our name.

**input** The `raw_input()` function accepts only text input:

```
>>> name = raw_input('Who\'s there ? ')
```

Displays `Who's there ?` on screen and assigns what the user types in to the variable `name`.

**output** With `name` in its argument, the `print` command displays the value of `name`:

```
>>> print 'hello ', name , '!'
```

*develop Python code interactively at the prompt*

# Interactive Python code

Let us write a program that asks for our name, as input.  
And then, as output, writes hello followed by our name.

**input** The `raw_input()` function accepts only text input:

```
>>> name = raw_input('Who\'s there ? ')
```

Displays **Who's there ?** on screen and assigns what the user types in to the variable `name`.

**output** With `name` in its argument, the `print` command displays the value of `name`:

```
>>> print 'hello ', name , '!'
```

*develop Python code interactively at the prompt*

# Interactive Python code

Let us write a program that asks for our name, as input.  
And then, as output, writes hello followed by our name.

**input** The `raw_input()` function accepts only text input:

```
>>> name = raw_input('Who\'s there ? ')
```

Displays **Who's there ?** on screen and assigns what the user types in to the variable `name`.

**output** With `name` in its argument, the `print` command displays the value of `name`:

```
>>> print 'hello ', name , '!'
```

*develop Python code interactively at the prompt*

# our first interactive program

running at the command prompt

Computer  
Architecture

Hardware  
Components

Programming  
Environments

Getting  
Started with  
Python

Installing Python

Executing Python  
code

Number  
Systems

Decimal and Binary  
notations

Running Sage

Summary +  
Assignments

The file `hello_there.py` contains

```
# L-2 MCS 260 an interactive program
name = raw_input('Who\'s there ? ')
print 'hello ', name , '!'
```

The `#` signs the start of a comment, the line following `#` is ignored by the interpreter.

At the command prompt, we type

```
python hello_there.py
```

On Windows, double clicking on the file with `.py` extension will execute the program.

# our first interactive program

running at the command prompt

Computer  
Architecture

Hardware  
Components

Programming  
Environments

Getting  
Started with  
Python

Installing Python

Executing Python  
code

Number  
Systems

Decimal and Binary  
notations

Running Sage

Summary +  
Assignments

The file `hello_there.py` contains

```
# L-2 MCS 260 an interactive program
name = raw_input('Who\'s there ? ')
print 'hello ', name , '!'
```

The `#` signs the start of a comment, the line following `#` is ignored by the interpreter.

At the command prompt, we type

```
python hello_there.py
```

On Windows, double clicking on the file with `.py` extension will execute the program.

# our first interactive program

running at the command prompt

Computer  
Architecture

Hardware  
Components

Programming  
Environments

Getting  
Started with  
Python

Installing Python

Executing Python  
code

Number  
Systems

Decimal and Binary  
notations

Running Sage

Summary +  
Assignments

The file `hello_there.py` contains

```
# L-2 MCS 260 an interactive program
name = raw_input('Who\'s there ? ')
print 'hello ', name , '!'
```

The `#` signs the start of a comment, the line following `#` is ignored by the interpreter.

At the command prompt, we type

```
python hello_there.py
```

On Windows, double clicking on the file with `.py` extension will execute the program.

# Computer Architecture first steps with Python

## Computer Architecture

Hardware Components

Programming Environments

## Getting Started with Python

Installing Python

Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

- 1 Computer Architecture  
Hardware Components  
Programming Environments
- 2 Getting Started with Python  
Installing Python  
Executing Python code
- 3 Number Systems**  
**Decimal and Binary notations**
- 4 Running Sage
- 5 Summary + Assignments

# Decimal Notation of Numbers

## Computer Architecture

Hardware Components  
Programming Environments

## Getting Started with Python

Installing Python  
Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

- The value of  $284 = 2 \times 10^2 + 8 \times 10^1 + 4 \times 10^0$ . 2,8,4 are the digits of the number, 10 is the base. The position of each digit determines its contribution to the value of the number.

- For any base  $B$ , a number  $n$  is denoted by  $m$  coefficients  $c_i$ ,  $i = m, m - 1, \dots, 1, 0$ ,  $0 \leq c_i < B$ :

$$n = c_m B^m + c_{m-1} B^{m-1} + \dots + c_1 B^1 + c_0 B^0.$$

- From base five to decimal notation:

$$\begin{aligned} 2104_5 &= 2 \times 5^3 + 1 \times 5^2 + 0 \times 5^1 + 4 \times 5^0 \\ &= 250 + 25 + 0 + 4 \\ &= 279_{10} \end{aligned}$$

# Decimal Notation of Numbers

## Computer Architecture

Hardware Components  
Programming Environments

## Getting Started with Python

Installing Python  
Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

- The value of  $284 = 2 \times 10^2 + 8 \times 10^1 + 4 \times 10^0$ .  
2,8,4 are the digits of the number, 10 is the base.  
The position of each digit determines its contribution to the value of the number.
- For any base  $B$ , a number  $n$  is denoted by  $m$  coefficients  $c_i$ ,  $i = m, m - 1, \dots, 1, 0$ ,  $0 \leq c_i < B$ :

$$n = c_m B^m + c_{m-1} B^{m-1} + \dots + c_1 B^1 + c_0 B^0.$$

- From base five to decimal notation:

$$\begin{aligned} 2104_5 &= 2 \times 5^3 + 1 \times 5^2 + 0 \times 5^1 + 4 \times 5^0 \\ &= 250 + 25 + 0 + 4 \\ &= 279_{10} \end{aligned}$$

# Decimal Notation of Numbers

## Computer Architecture

Hardware Components  
Programming Environments

## Getting Started with Python

Installing Python  
Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

- The value of  $284 = 2 \times 10^2 + 8 \times 10^1 + 4 \times 10^0$ .  
2,8,4 are the digits of the number, 10 is the base.  
The position of each digit determines its contribution to the value of the number.
- For any base  $B$ , a number  $n$  is denoted by  $m$  coefficients  $c_i$ ,  $i = m, m - 1, \dots, 1, 0$ ,  $0 \leq c_i < B$ :

$$n = c_m B^m + c_{m-1} B^{m-1} + \dots + c_1 B^1 + c_0 B^0.$$

- From base five to decimal notation:

$$\begin{aligned} 2104_5 &= 2 \times 5^3 + 1 \times 5^2 + 0 \times 5^1 + 4 \times 5^0 \\ &= 250 + 25 + 0 + 4 \\ &= 279_{10} \end{aligned}$$

# Binary Numbers

## Computer Architecture

Hardware  
Components

Programming  
Environments

## Getting Started with Python

Installing Python  
Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

- The base is two, the coefficients are bits  $\in \{0, 1\}$ .

- The first 16 natural numbers — need 4 bits:

0000 = 0    0001 = 1    0010 = 2    0011 = 3

0100 = 4    0101 = 5    0110 = 6    0111 = 7

1000 = 8    1001 = 9    1010 = A    1011 = B

1100 = C    1101 = D    1110 = E    1111 = F

The hexadecimal ‘digits’ are 0,1,2,...,9,A,B,C,D,E,F.

- It is straightforward to convert binary into hexadecimal and hexadecimal into binary numbers.

# Binary Numbers

## Computer Architecture

Hardware Components

Programming Environments

## Getting Started with Python

Installing Python

Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

- The base is two, the coefficients are bits  $\in \{0, 1\}$ .

- The first 16 natural numbers — need 4 bits:

0000 = 0    0001 = 1    0010 = 2    0011 = 3

0100 = 4    0101 = 5    0110 = 6    0111 = 7

1000 = 8    1001 = 9    1010 = A    1011 = B

1100 = C    1101 = D    1110 = E    1111 = F

The hexadecimal 'digits' are 0,1,2,...,9,A,B,C,D,E,F.

- It is straightforward to convert binary into hexadecimal and hexadecimal into binary numbers.

# Binary Numbers

## Computer Architecture

Hardware Components

Programming Environments

## Getting Started with Python

Installing Python

Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

- The base is two, the coefficients are bits  $\in \{0, 1\}$ .

- The first 16 natural numbers — need 4 bits:

0000 = 0    0001 = 1    0010 = 2    0011 = 3

0100 = 4    0101 = 5    0110 = 6    0111 = 7

1000 = 8    1001 = 9    1010 = A    1011 = B

1100 = C    1101 = D    1110 = E    1111 = F

The hexadecimal ‘digits’ are 0,1,2,...,9,A,B,C,D,E,F.

- It is straightforward to convert binary into hexadecimal and hexadecimal into binary numbers.

# Binary Numbers

## Computer Architecture

Hardware Components

Programming Environments

## Getting Started with Python

Installing Python

Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

- The base is two, the coefficients are bits  $\in \{0, 1\}$ .

- The first 16 natural numbers — need 4 bits:

0000 = 0    0001 = 1    0010 = 2    0011 = 3

0100 = 4    0101 = 5    0110 = 6    0111 = 7

1000 = 8    1001 = 9    1010 = A    1011 = B

1100 = C    1101 = D    1110 = E    1111 = F

The hexadecimal ‘digits’ are 0,1,2,...,9,A,B,C,D,E,F.

- It is straightforward to convert binary into hexadecimal and hexadecimal into binary numbers.

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B_{16}$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

# Converting Numbers

from decimal to binary

Convert 123 into binary format:

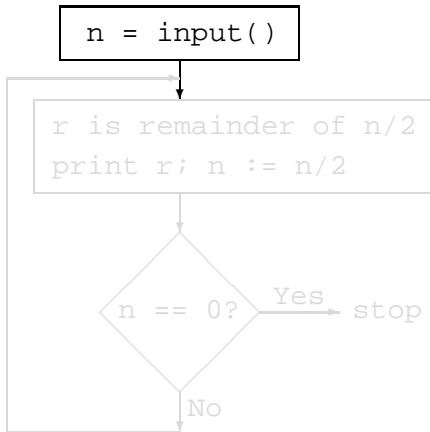
$n$	$n/2$	$n \bmod 2$	
123	61	1	$123 = 61 \times 2 + 1$
61	30	1	$61 = 30 \times 2 + 1$
30	15	0	$30 = 15 \times 2 + 0$
15	7	1	$15 = 7 \times 2 + 1$
7	3	1	$7 = 3 \times 2 + 1$
3	1	1	$3 = 1 \times 2 + 1$
1	0	1	$1 = 0 \times 2 + 1$

$$\begin{aligned}
 123 &= 1 + 2 \times 61 = 1 + 2 \times (1 + 2 \times 30) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times 15)) \\
 &= 1 + 2 \times (1 + 2 \times (0 + 2 \times (1 + 2 \times 7))) \\
 &= \dots
 \end{aligned}$$

So  $123 = 1111011 = 7B$ .

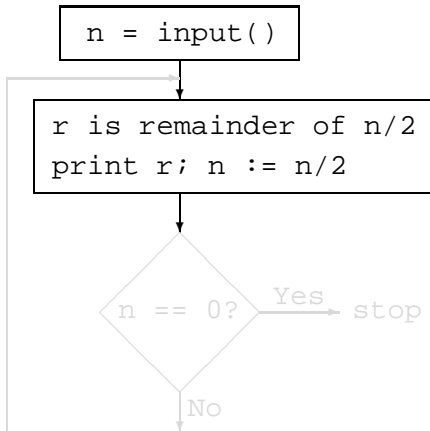
# Flowchart

## conversion algorithm



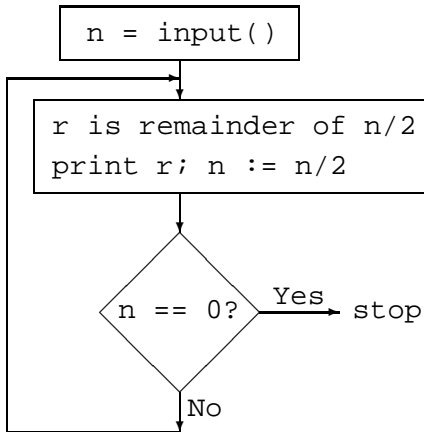
# Flowchart

conversion algorithm



# Flowchart

conversion algorithm



# Running Sage

## Ways to compute with Sage:

- a notebook blends commands with commentary
  - graphical user interface
  - runs in a browser
- command line use if no graphical output is needed and more dedicated to computationally intensive jobs
- language is Python, with some variation  
Python (**version < 3.0**):

```
>>> 68/25  
2
```

Sage:

```
sage: 68/25  
68/25
```

# Running Sage

## Ways to compute with Sage:

- a notebook blends commands with commentary
  - graphical user interface
  - runs in a browser
- command line use if no graphical output is needed and more dedicated to computationally intensive jobs
- language is Python, with some variation  
Python (**version < 3.0**):

```
>>> 68/25  
2
```

Sage:

```
sage: 68/25  
68/25
```

# Running Sage

## Ways to compute with Sage:

- a notebook blends commands with commentary
  - graphical user interface
  - runs in a browser
- command line use if no graphical output is needed and more dedicated to computationally intensive jobs
- language is Python, with some variation  
Python (**version < 3.0**):

```
>>> 68/25  
2
```

Sage:

```
sage: 68/25  
68/25
```

# Running Sage

## Ways to compute with Sage:

- a notebook blends commands with commentary
  - graphical user interface
  - runs in a browser
- command line use if no graphical output is needed and more dedicated to computationally intensive jobs
- language is Python, with some variation  
Python (**version < 3.0**):

```
>>> 68/25  
2
```

Sage:

```
sage: 68/25  
68/25
```

# Number Systems

with Python and Sage

Hexadecimal conversions in Python with the % operator:

```
>>> "%X" % 123
```

```
'7B'
```

```
>>> '%x' % 123
```

```
'7b'
```

The same calculations are possible in Sage.

Sage has much more refined number systems:

- 1 integers and modular arithmetic
- 2 exact calculations with rational numbers
- 3 arbitrary precision float and complex
- 4 algebraic numbers

# Number Systems

with Python and Sage

Hexadecimal conversions in Python with the % operator:

```
>>> "%X" % 123
```

```
'7B'
```

```
>>> '%x' % 123
```

```
'7b'
```

The same calculations are possible in Sage.

Sage has much more refined number systems:

- 1 integers and modular arithmetic
- 2 exact calculations with rational numbers
- 3 arbitrary precision float and complex
- 4 algebraic numbers

# Summary + Assignments

## Computer Architecture

Hardware Components  
Programming Environments

## Getting Started with Python

Installing Python  
Executing Python code

## Number Systems

Decimal and Binary notations

## Running Sage

## Summary + Assignments

### Recommended reading:

- Python tutorial on <http://docs.python.org/tut/tut.html>
- sections 1.4 & 1.5 of *Computer Science. An Overview*
- chapter 1 of *Python Programming in Context*

### Assignments:

- 1 Given the base and a sequence of coefficients of a number, give the algorithm to evaluate the number.
- 2 Write down pseudocode for the algorithm to compute the binary representation of a number.
- 3 Compute examples of general number conversions from any base to any other base.
- 4 What is the algorithm for such general conversions?