Introduction to Computer Science and Programming

Lecture 3

Yue Zhang
Westlake University

August 1, 2023

WestlakeNLP

Chapter 3.

Inputs and Outputs

WestlakeNLP

Overview

WestlakeNLP

Overview

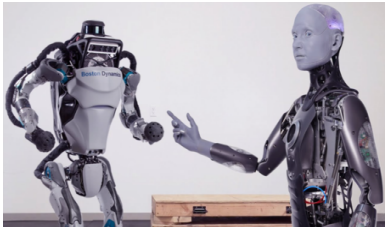
WestlakeNLP

Analogue Signals

Pictures

Videos

Robot Action



Sounds

WestlakeNLP

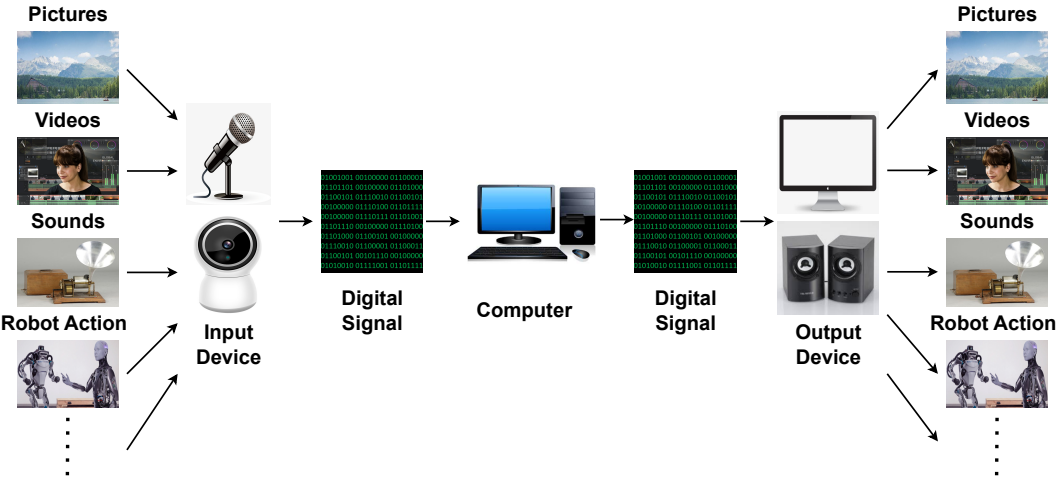
Analogue Signals

Figure: The Matrix

WestlakeNLP

Analogue Signals and Computers

- How does your computer interact with the physical world?

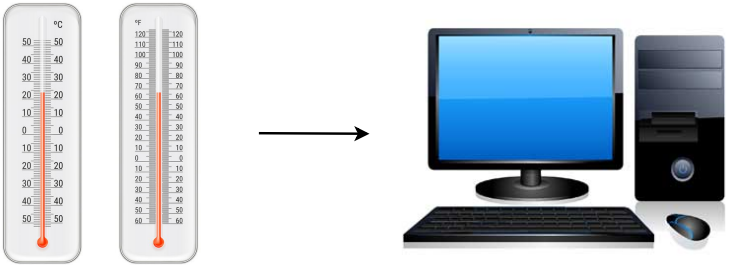


Turning Different Things into Binary Numbers

- Numbers
- Text
- Audio Signals
- Images
- Video Signals

WestlakeNLP

Digitizing Numbers



- Turn real numbers to binary numbers.
- Make sure that they are stored in words.

WestlakeNLP

Digitizing Integers

Binary for 19?

$$19 = 16 + 2 + 1 = 2^4 + 2^1 + 2^0 = 10011$$

$19 \div 2 = 9 \cdots\cdots 1$ ↑

$9 \div 2 = 4 \cdots\cdots 1$

$4 \div 2 = 2 \cdots\cdots 0$ read

$2 \div 2 = 1 \cdots\cdots 0$

$1 \div 2 = 0 \cdots\cdots 1$

WestlakeNLP

Byte and Word

- Computers cannot store arbitrarily large numbers.
  - A binary digit is a bit.
  - 8 bits is a byte.
  - Computer organize numbers and other data in words.

32-bit machines: 1 word = 4 bytes = 32 bits
64-bit machines: 1 word = 8 bytes = 64 bits

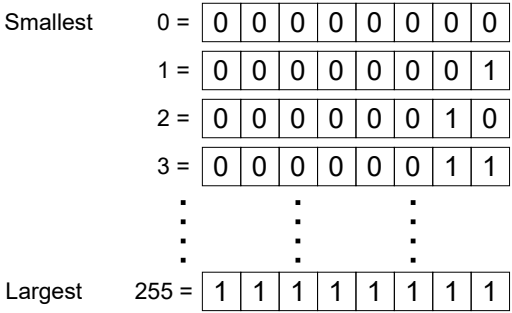
WestlakeNLP

Digitizing Integers

How many numbers can a computer word hold?

8-bit machines

$2^8 = 256$



WestlakeNLP

Digitizing Integers

Machine Type	Capacity	Value Range
8-bit machines	$2^8 = 256$	0 ~ 255
16-bit machines	$2^{16} = 65536$	0 ~ 65535
32-bit machines	2^{32}	0 ~ 2^{32}-1
64-bit machines	2^{64}	0 ~ 2^{64}-1

WestlakeNLP

How to represent negative numbers?

- in 8-bit machines

100 ⟶ 0 0 0 0 0 1 0 0

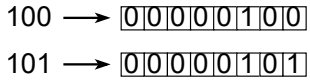
101 ⟶ 0 0 0 0 0 1 0 1

What about -100, -101?

WestlakeNLP

How to represent negative numbers?

- in 8-bit machines



What about -100, -101?
the negative sign?

WestlakeNLP

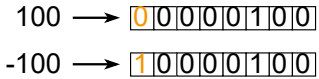
How to represent negative numbers?

- Using the first bit to indicate sign.

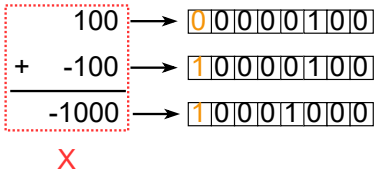
WestlakeNLP

How to represent negative numbers?

- Using the first bit to indicate sign.



- Not friendly for arithmatic.



WestlakeNLP

How to represent negative numbers?

- 2's complement

  first reverse every bit,
  second add one.

e.g. 100 ⟶ 0 0 0 0 0 1 0 0

reverse ⟶ 1 1 1 1 1 0 1 1

add 1 ⟶ 1 1 1 1 1 1 0 0 -100

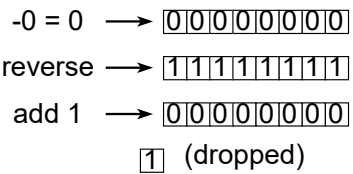
still requires the first bit as sign.

WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic

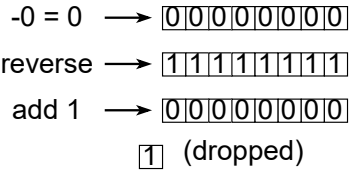


WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic



WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic

-100+100=0

		1	1	1	1	1	1	0	0
+		0	0	0	0	0	1	0	0

WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic

-100+100=0

		1	1	1	1	1	1	0	0
+		0	0	0	0	0	1	0	0
									0

WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic

-100+100=0

		1	1	1	1	1	1	0	0
+		0	0	0	0	0	1	0	0
							0	0

WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic

-100+100=0

		1	1	1	1	1	1	0	0
+		0	0	0	0	0	1	0	0
							0	0	0

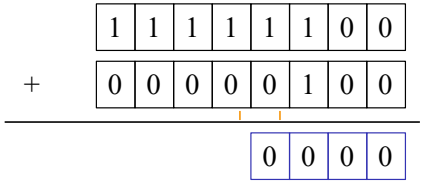
WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic

-100+100=0



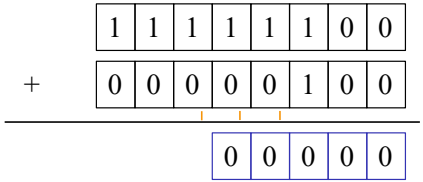
WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic

-100+100=0



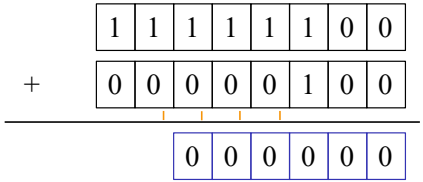
WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic

-100+100=0



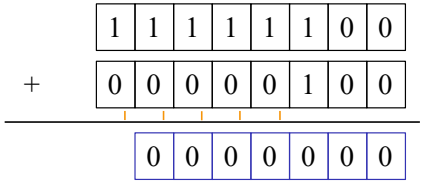
WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic

-100+100=0



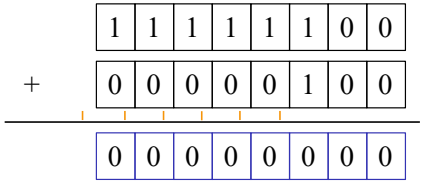
WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic

-100+100=0



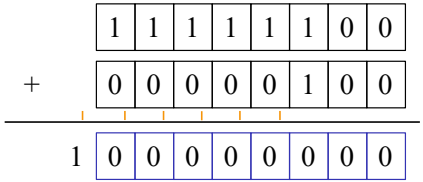
WestlakeNLP

How to represent negative numbers?

- 2's complement

  advantage in arithmatic

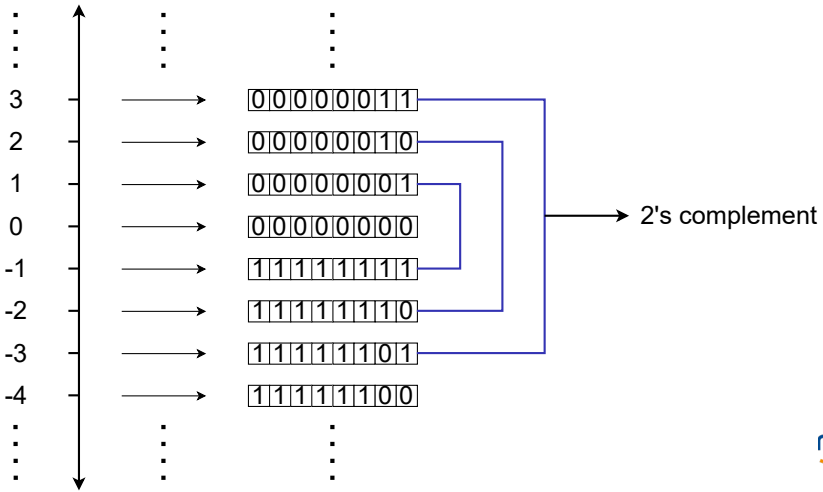
-100+100=0



WestlakeNLP

Digitizing Integers

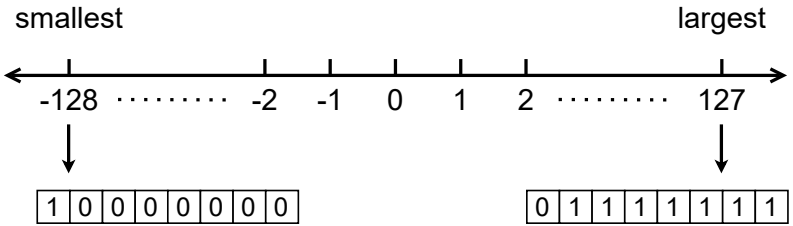
- Negative Numbers



WestlakeNLP

Digitizing Integers

- Negative Numbers



- Range: $-128 \sim 127$

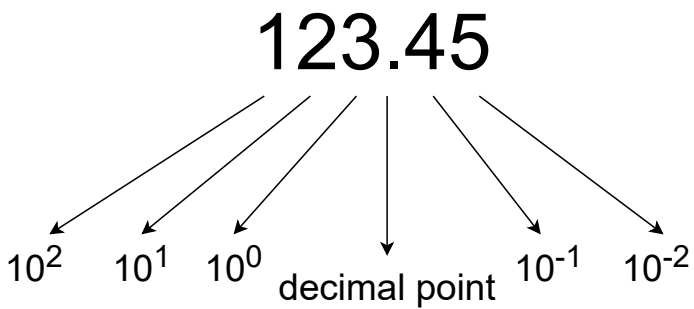
WestlakeNLP

Digitizing Integers

Machine Type	Capacity	Non-negative Range	Range
8-bit machines	$2^8 = 256$	0 ~ 255	-128 ~ 127
16-bit machines	$2^{16} = 65536$	0 ~ 65535	-32768 ~ 32767
32-bit machines	2^{32}	0 ~ 2^{32}-1	-2^{31} ~ 2^{31}-1
64-bit machines	2^{64}	0 ~ 2^{64}-1	-2^{63} ~ 2^{63}-1

WestlakeNLP

Digitizing Real Numbers



WestlakeNLP

Digitizing Real Numbers

$$123.45 = 12345 \times 10^{-2} \ (12345\text{E} - 2)$$

General Form: sign mantissa $\times 10^{exp}$

WestlakeNLP

Digitizing Real Numbers

$$123.45 = 12345 \times 10^{-2} \ (12345\text{E} - 2)$$

General Form: sign mantissa $\times 10^{exp}$

e.g.,	+	12345	-2	$\rightarrow$	123.45
e.g.,	-	123	6	$\rightarrow$	123000000

WestlakeNLP

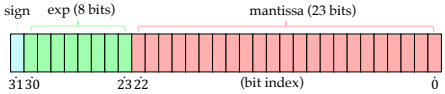
Digitizing Real Numbers

sign mantissa $\times 10^{exp}$

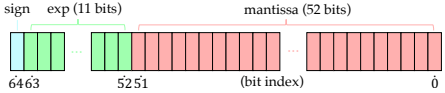
e.g., $+ 12345 \times 10^{-2}$

also, + 12345 E -2

32-bit machines 1 bit 8 bits 23bits



64-bit machines 1 bit 11 bits 52bits



WestlakeNLP

Digitizing Real Numbers

- Overflow and underflow
  - 123 E 65536?
  - 0.00000000000000000000001?

WestlakeNLP

Digitizing Text

- Map Each Character into a Byte (ASCII)

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

e.g.,

- 'A' → #65
- 'a' → #97

WestlakeNLP

Digitizing Text

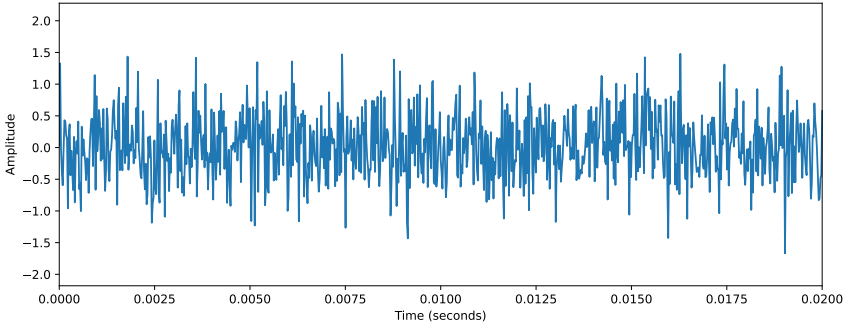
- For alphabets over 255 characters?
  Chinese? (GB2312, GBK)
  UNICODE – Map each character into 2 Bytes.

Code (Hex)	Character	Source
0041	A	English (Latin)
042F	Я	Russian (Cyrillic)
0E09	ฉ	Thai
13EA	Ꮺ	Cherokee
211E	℞	Letterlike symbols
21CC	⇌	Arrows
282F	⠯	Braille
345F	㑟	Chinese/Japanese/ Korean (common)

WestlakeNLP

Digitizing Audio

Soundwave is analogue signal.

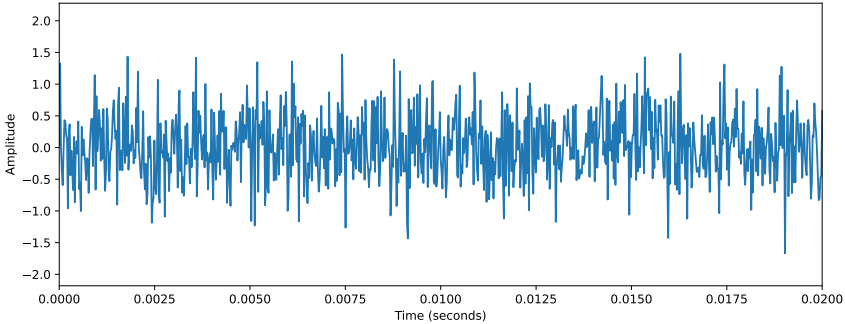


Radio process such signals using circuits.

WestlakeNLP

Digitizing Audio

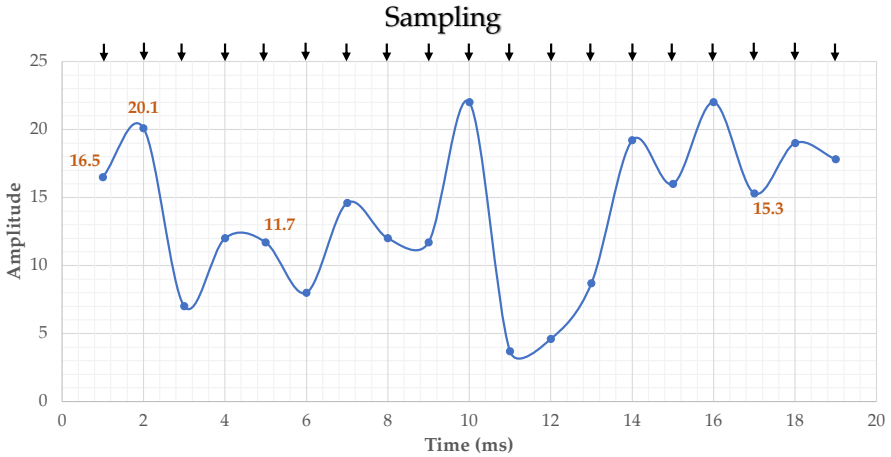
Two things to Digitize.



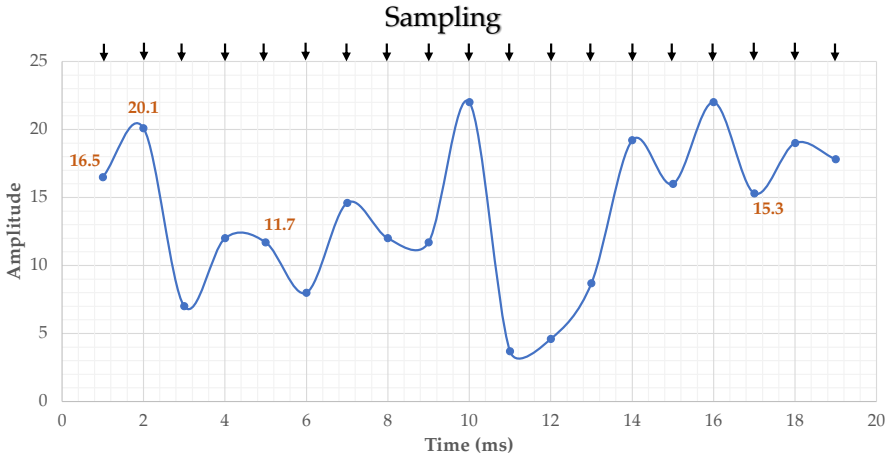
Amplitude & time

WestlakeNLP

Digitizing Audio



Digitizing Audio



Sampling Frequency --- 40,000 times per second.

Digitizing Image – Grey Scale Image

165	243	139	114	136	122	236	145
241	210	235	242	58	226	236	163
163	53	198	142	149	70	16	135
58	60	46	19	109	126	95	83
220	202	189	128	112	254	148	190
42	84	215	159	218	174	235	148
203	46	42	99	82	102	33	233
11	107	60	10	97	79	58	130

Digitizing Image – Grey Scale Image

- Pixels and Resolution

  1024×768
  1920×1024

  The Granularity of Samples

WestlakeNLP

Digitizing Image – Color Image

Each pixel represented with RGB (red-green-blue) values.

Red	Green	Blue	Color	
0	0	0	Black	
255	255	255	White	
255	255	0	Yellow	
255	192	203	Pink	
165	42	42	Brown	
160	32	240	Purple	
176	48	96	Maroon	

WestlakeNLP

Digitizing Video – Segvene of Images

- Time and Space Resolutions

WestlakeNLP

Compressing Data

- Signals can be large.

Modal	Lossless Compression	Lossy Compression
Audio	APE, FLAC, TTA	MP3, WMA, OGG
Image	PNG, TIFF, BMP	JPEG, ICER, DjVu

WestlakeNLP

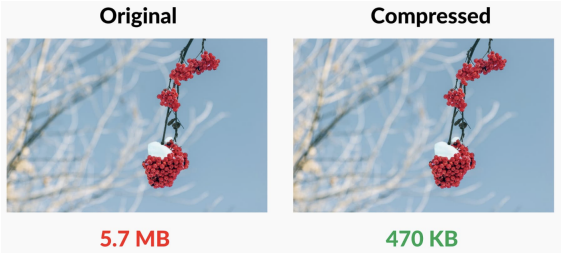
Compressing Data

Lossless Compression

- A String Compression Case
  - $aaaabbccc \rightarrow a4b2c3$

Lossy Compression

- Lose Less Important Information



WestlakeNLP

Overview

WestlakeNLP

Python Programs

- A stand-alone program can be launched from the OS.

```
Yues~MacBook~Pro:Desktop$ python savings.py
[Savings Calculator]
    Enter interest rate: 0.036
    Enter initial sum: 10000
    Enter number of years: 5
    After 5 years, the savings will be 11934.35.
Yues~MacBook~Pro:Desktop$
```

WestlakeNLP

Python Programs

- We already know how to do math computations.
- Need to achieve console input and output.

```
Yues~MacBook~Pro:Desktop$ python savings.py
[Savings Calculator]
    Enter interest rate: 0.036
    Enter initial sum: 10000
    Enter number of years: 5
    After 5 years, the savings will be 11934.35.
Yues~MacBook~Pro:Desktop$
```

WestlakeNLP

String – String Literals

```
>>> s='abc'
>>> s
'abc'
>>> type(s)
<class 'str'>
```

- Single Quote

```
>>> s="he said:'hello'"
>>> s
"he said:'hello'"
>>> type(s)
<class 'str'>
```

- Double Quotes

WestlakeNLP

String – String Literals

```
>>> s="""
... abc
... def
... ghi
... """
>>> s
'\nabc\ndef\nghi\n'
>>> type(s)
<class 'str'>
```

```
>>> s='abc	def	ghi'
>>> s
'abc\tdef\tghi'
```

- Three Quotes

- **Escaped Form**
  - return → "\n"
  - tab → "\t"

WestlakeNLP

String – String Operators

```
>>> s1='abc'
>>> s2='def'
>>> s1+s2
'abcdef'
```

- Concatenation (+)

```
>>> s='abc'
>>> s*3
'abcabcabc'
```

- Repetition (*)

- Operators + and *
  - **Polymorphism** for different operator types, the evaluation is different.

WestlakeNLP

String – String Operators

```
>>> s='abc'
>>> s[0]
'a'
>>> s[1]
'b'
>>> s[2]
'c'
```

```
>>> i=2
>>> s[i]
'c'
```

- Slice ([ ])
  - the getitem operator
  - index start from 0
- Operands can of course be variables.

WestlakeNLP

String – String Operators

```
>>> s='abc'
>>> i=-1
>>> s[-1]
'c'
>>> i=i-1
>>> s[i]
'b'
>>> i-=1
>>> s[i]
'a'
```

- Right-to-left getitem index

WestlakeNLP

String – String Operators

```
>>> s='abc'
>>> s[3]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: string index out of range
>>>
>>> s[-4]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: string index out of range
```

- Out of Range
  - Evaluating the expression results in Error objects.

WestlakeNLP

String – String Operators

```
>>> s='abc'
>>> s[1:2]
'b'
>>> s[0:2]
'ab'
>>> s[1:3]
'bc'
>>> s[-2:-1]
'b'
```

- Range Slice ([:])
  - substrings
    - The getslice operator takes more arguments.
- [startIndex:endIndex+1]

WestlakeNLP

String – String Operators

- Special Cases

```
>>> s='abc'
>>> s[2:2]       # -> empty string
''
>>> s[2:1]       # -> empty string
''
>>> s[2:5]       # -> out of bounding
'c'
>>> s[1:]        # -> default end
'bc'
>>> s[:2]        # -> default start
'ab'
>>> s[:]
'abc'
```

WestlakeNLP

String – Type Conversion

```
>>> i=123
>>> s=str(i)
>>> s
'123'
```

- Integer to String

```
>>> f=1.234
>>> s=str(f)
>>> s
'1.234'
>>> type(s)
<class 'str'>
```

- Float to String

WestlakeNLP

String – Type Conversion

```
>>> s='123'
>>> int(s)
123
```

- String to Integer

```
>>> s='1.234'
>>> f=float(s)
>>> f
1.234
>>> type(f)
<class 'float'>
```

- String to Float

```
>>> i=int(s)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int()
    with base 10: '1.234'
```

- Error Value

WestlakeNLP

String – String Formatting

- Pattern Strings

```
>>> s='%d'%123                          # %d -- integer
>>> s                                   # 123 after % -- fills %d
'123'
>>> s='The integers:%d'%123
>>> s
'The integers:123'

>>> '%5d'%123                           # 5 spicifies string size
'  123'

>>> '%-5d'%123                          # '-' means align to the left
'123  '

>>> 'abc%-5dd'%123
'abc123  d'
```

String – String Formatting

```
>>> '%f'%1.23
'1.230000'
>>> '%5.2f'%1.234            # '.2' -- string size precision
' 1.23'
>>> '%.2f'%1.234
'1.23'
>>> 'abc%.2f'%(1.0/7)
'abc0.14'
```

WestlakeNLP

String – String Formatting

- Multiple Patterns

```
>>> 'A string with one integer %d, followed by one floating
    point number %.2f'%(123,1.234)
'A string with one integer 123, followed by one floating
    point number 1.23'
```

WestlakeNLP

String

- Checkout the binary encoding of strings.

```
>>> ord('a')
97
>>> ord('A')
65
>>> ord('z')
122
>>> chr(97)
'a'
>>> chr(98)
'b'
```

- ord – get ASCII code
  - from character
- chr – get character
  - from ASCII code

WestlakeNLP

Text IO – print

- The Print Function

```
>>> print(123)                    # integer
123
>>> print(1.234)                  # float
1.234
>>> print(3+5-2*6)                # expression evaluated
-4
>>> print('abc')
abc
>>> print('abc\ndef\nxyz')        # strings have print formats
abc
def
xyz
>>> print('abc\\n')               # escape format for '\'
abc\n
```

Text IO – input

- The Input Function

```
>>> s=input()
abc
>>> s                              # return a string from user
'abc'
>>> s=input("Enter a number: ")   # takes a prompt argument
Enter a number:
```

WestlakeNLP

Text IO – input

- The Input Function

```
>>> s=input()
abc
>>> s                                  # return a string from user
'abc'
>>> s=input("Enter a number: ")  # takes a prompt argument
Enter a number: 123
>>> s
'123'
>>> int(s)
123
```

WestlakeNLP

A Stand-Alone Python Program

- savings.py

```
print('[Savings Calculator]')
rate = float(input('  Enter interest rate: '))
initial_sum = int(input('  Enter initial sum: '))
number_of_years = int(input('  Enter number of years: '))

final_sum = initial_sum*(1.0+rate)**number_of_years

print("    After %d years, the savings will be %.2f."%(
    number_of_years, final_sum))
```

WestlakeNLP

A Stand-Alone Python Program

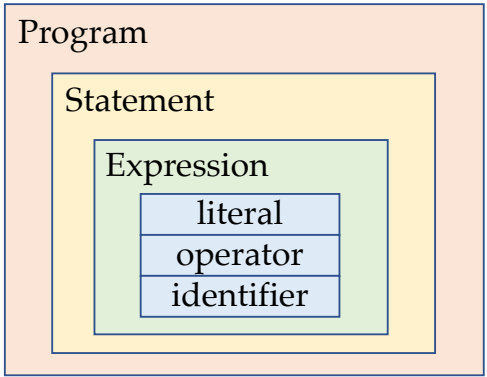
- Save and execute it.

```
Yues~MacBook~Pro:Desktop$ python savings.py
[Savings Calculator]
    Enter interest rate: 0.036
    Enter initial sum: 10000
    Enter number of years: 5
    After 5 years, the savings will be 11934.35.
Yues~MacBook~Pro:Desktop$
```

WestlakeNLP

Python Programs

- Building Blocks



WestlakeNLP

Python Programs

- Summary of Types
  int, float, str, function, module, type
- Summary of Operators
  +, -, *, /, **, %, ., [], [:]
- Summary of Statement
  assignment(=,+=,...), import, print, input

WestlakeNLP

Python Programs

- in line comments (start with #)

```
# show function
print('[Savings Calculator]')
```

- end of line comments (start with #)

```
# get inputs
rate = float(input('  Enter interest rate: ')) # initial rate
initial_sum = int(input('  Enter initial sum: ')) # initial sum
number_of_years = int(input('  Enter number of years: ')) # initial years

# calculate final sum
final_sum = initial_sum*(1.0+rate)**number_of_years # final sum of savings
```

- Comments makes the program more readable.

```
# output
print("    After %d years, the savings will be %.2f."%(number_of_years, final_sum))
```

Programs VS. Human Languages

- operate a computer / communicate with human
- formal language vs. natural language
- more mathmatical
- more rigid syntax
- limited semantics
- restricted vocabulary

WestlakeNLP

This week check-off:

Simple Programs

WestlakeNLP