SBS
Biohackathon

BIT by BIT

Python Workshop 1

Instructors: Sundara & Min Qi

WORKSHOP CONTENT

SBS
Biohackathon

INTRO TO BITS

What is Binary Code (Bit)?

- The bit is the most basic unit of information in computing and digital communications.
    - One Integer uses 4 Bytes
    - 1 Byte = 8 bits
    - One Integer uses 32 bits to store in the computer’s memory
- The bit represents a logical state with one of two possible values.
- These values are most commonly represented as either "1" or "0", there's other representations such as true/false, yes/no
- Since each digit in binary can have 2 values, the base is 2

Properties of binary code

- Only two possible values: 1 or 0
    - Think of it like a switch (On or Off)
- Each digit position represents an increasing power of 2

Functions: For flagging system; on/off switch; Storage etc.

Properties of binary code

- Each digit position represents an increasing power of 2

1	0	0	1	0	1	1	0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

$= 128 + 16 + 4 + 2$

$= 150$

What is Binary Code (Bit)?

- Practice! Convert the following binary to integer form

$10101 = 2^4 + 2^2 + 2^0 = 21$

$100011 = 2^5 + 2^1 + 2^0 = 35$

$11000 = 2^4 + 2^3 = 24$

What is Binary Code (Bit)?

- Example:

$28 = 11100$

$= 1 \quad 1 \quad 1 \quad 0 \quad 0$

$= 2^4 + 2^3 + 2^2$

$= 16 + 8 + 4$

$47 = 101111$

$= 1 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1$

$= 2^5 + 2^3 + 2^2 + 2^1 + 2^0$

$= 32 + 8 + 4 + 2 + 1$

What is Binary Code (Bit)?

- Practice! Convert the following decimal numbers to binary
- 18
- 36
- 57

What is Binary Code (Bit)?

- ANSWERS!

$$18 = 2^4 + 2^1$$
$$= 10010$$

$$36 = 2^5 + 2^2$$
$$= 100100$$

$$57 = 2^5 + 2^4 + 2^3 + 2^0$$
$$= 111001$$

Integer to Binary Conversion in python

- You can convert integers to binary in python using the **bin()** function

- Also you can convert binary to integers using the **int()** function

SBS
Biohackathon

Bitwise Operators

What are Bitwise Operators

- You can use bitwise operators to perform Boolean logic on individual bits.
- Similar to python Operators (+, -, % etc.) but specifically to compare binary numbers

Bitwise AND

- Denoted by: &
- For each pair of bits occupying the **same position** in the two numbers, it **returns a 1 only when both bits are 1**
- Example:

  a = 11001

  b = 10101

a & b = 10001

Bitwise OR

- Denoted by: |
- Seen as a union of arguments
- 1 + 1 = 1 AND 0 +1 = 1 BUT 0 + 0 =0
- Example:

  a = 11001

  b = 10101

a | b = 11101

Bitwise Left Shift

- Denoted by: <<
- Moves the bits of its first operand to the left
- Shifting a single bit to the left by one place doubles its value.
- Example:

$$a = 10011_2 = 19_{10}$$
$$a << 1 = 100110_2 = 38_{10}$$
$$a << 2 = 1001100_2 = 76_{10}$$

Bitwise Right Shift

- Denoted by: >>
- Analogous to left shift; moves the bits of its first operand to the right
- Shifting a single bit to the right; dropping the rightmost bit
- Example:

$$a = 1001101_2 = 77_{10}$$
$$a >> 1 = 100110_2 = 38_{10}$$
$$a >> 2 = 10011_2 = 19_{10}$$

Bitwise Operators Summary

Operator	Name	Description
&	AND	Sets each bit to1 only if both bits are 1
|	OR	Sets each bit to 1 if one of two bits is 1
<<	LEFT SHIFT	Shift left by pushing zeros in from the right and let the leftmost bits fall off
>>	RIGHT SHIFT	Shift right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off

SBS
Biohackathon

Shift – Or Algorithm

Bitap / Shift-Or Algorithm: What is it

- String matching algorithm to find **approximate matches** to a given pattern queried in a text

- Applications:
  - Mutations (insertions/deletions/substitutions) in DNA/RNA sequence
  - Identification of Biomarkers
  - Proteogenomic mapping

Bitap / Shift-Or Algorithm: Step by Step

01 Create Bitmask

02 Initialize Bitarray

03 Match letters in text to pattern using bitmask and bitarray (Exact String Matching)

04 Inexact String Matching

Step 1: Create bitmask

- A bitmask is a binary string constructed based on whether a specific letter in the text is found in the pattern

Pattern: ATC
Sequence: ATCGATC

$$B[T_j]$$

Bits:
1: mismatch
0: match

	b_1	b_2	b_3	bitmask
	A	T	C	
A	0	1	1	0b011
G	1	1	1	0b111
T	1	0	1	0b101
C	1	1	0	0b110
*	1	1	1	0b111

Step 2: Initialize bitarray

- The bitarray, D, is used like a placeholder to capture matches of each letter in the text to the queried pattern
- Binary string of the same length as queried pattern

Pattern: ATC
Sequence: ATCGATC

$$D = 0b111$$

Step 3: String Matching | Exact Matching

Pattern: ATC
Sequence: ATCGATC

A	T	C	G	A	T	C

D	T_j	$D >> 1$	$B[T_j]$	$(D >> 1) \mid B[T_j]$
111	A			

	b_3	b_2	b_1
$D >> 1$	0	1	1
$B[A]$	0	1	1
$(D >> 1) \mid B[A]$	0	1	1

Most significant bit

Step 3: String Matching | Exact Matching

Pattern: ATC
Sequence: ATCGATC

A	T	C	G	A	T	C

	$B[T_j]$
A	011
T	101
C	110
G	111
*	111

D	T_j	$D >> 1$	$B[T_j]$	$(D >> 1) \mid B[T_j]$
111	A	011	011	011
	T			
	C			

$(D >> 1) \mid B[C]$ **110**

Most significant bit

Match is found

Step 3: String Matching | Exact Matching
Try It Yourself!

Pattern: ATC
Sequence: ATCGATC

A	T	C	G	A	T	C

	$B[T_j]$
A	011
T	101
C	110
G	111
*	111

D	T_j	$D >> 1$	$B[T_j]$	$(D >> 1) \mid B[T_j]$
111	A	011	011	011
011	T	001	101	101
101	C	010	110	110
110	G			
	A			
	T			
	C			

Step 3: String Matching | Exact Matching
Try It Yourself! - Answers

Pattern: ATC
Sequence: ATCGATC

A	T	C	G	A	T	C

T_j	$(D >> 1) \mid B[T_j]$
A	011
T	101
C	110
G	111
A	011
T	101
C	110

	A	T	C	G	A	T	C
A	0	1	1	1	0	1	1
T	1	0	1	1	1	0	1
C	1	1	0	1	1	1	0

Step 4: String Matching | Inexact Matching

Pattern: ATC
Sequence: ATCGATC

A	T	C	G	A	T	C

C	T	C	G	A	T	C

How do we capture this hit?

Step 4: String Matching | Inexact Matching

- Perform exact string matching
- Repeat above steps but execute additional steps where:

1. $(\boldsymbol{D} >> 1) \mid \boldsymbol{B}[\boldsymbol{Tj}]$ from previous (exact) string matching right shift 1
2. Apply AND conditional to current $(\boldsymbol{D} >> 1) \mid \boldsymbol{B}[\boldsymbol{Tj}]$ and right-shifted $(\boldsymbol{D} >> 1) \mid \boldsymbol{B}[\boldsymbol{Tj}]$ of previous string match

Step 4: String Matching | Inexact Matching

Pattern: ATC
Sequence: ATCGATC

C	T	C	G	A	T	C

	$B[T_j]$
A	011
T	101
C	110
G	111
*	111

D	T_j	$D >> 1$	$B[T_j]$	$(D >> 1) \mid B[T_j]$
111	C	011	110	111
111	T	011	101	111
111	C	011	110	111

No hit captured

Step 4: String Matching | Inexact Matching

Pattern: ATC
Sequence: ATCGATC

C	T	C	G	A	T	C

	$B[T_j]$
A	011
T	101
C	110
G	111
*	111

D	T_j	$D >> 1$	$B[T_j]$	$(D >> 1) \mid B[T_j]$	$(D >> 1) \mid B[T_j]$ From previous exact search	$((D >> 1) \mid B[T_j]) >> 1$	$(((D >> 1) \mid B[T_j]) >> 1)$ & $(D >> 1) \mid B[T_j]$
111	C	011	110	111	111	011	011
011	T	001	101	101	111	011	001
001	C	000	110	110	111	011	010

Hit captured

Step 4: String Matching | Inexact Matching

Pattern: ATC
Sequence: ATCGATC

C	T	C	G	A	T	C

T_j	$(((D >> 1) \mid B[T_j]) >> 1) \& (D >> 1) \mid B[T_j]$
C	011
T	001
C	010

	C	T	C
A	0	0	0
T	1	0	1
C	1	1	0

- Red Font – One mismatch allowed

Step 4: Exact vs Inexact String matching

Pattern: ATC
Sequence: ATCGATC

Exact String Search

C	T	C	G	A	T	C

OR Conditional

1	0	1
0	1	
1	1	
0	0	= 0

Stricter

Inexact String Search

C	T	C	G	A	T	C

AND Conditional

1	0	0
0	1	
0	0	
1	1	= 1

More Lenient

SBS
Biohackathon

Shift – Or Sample Code