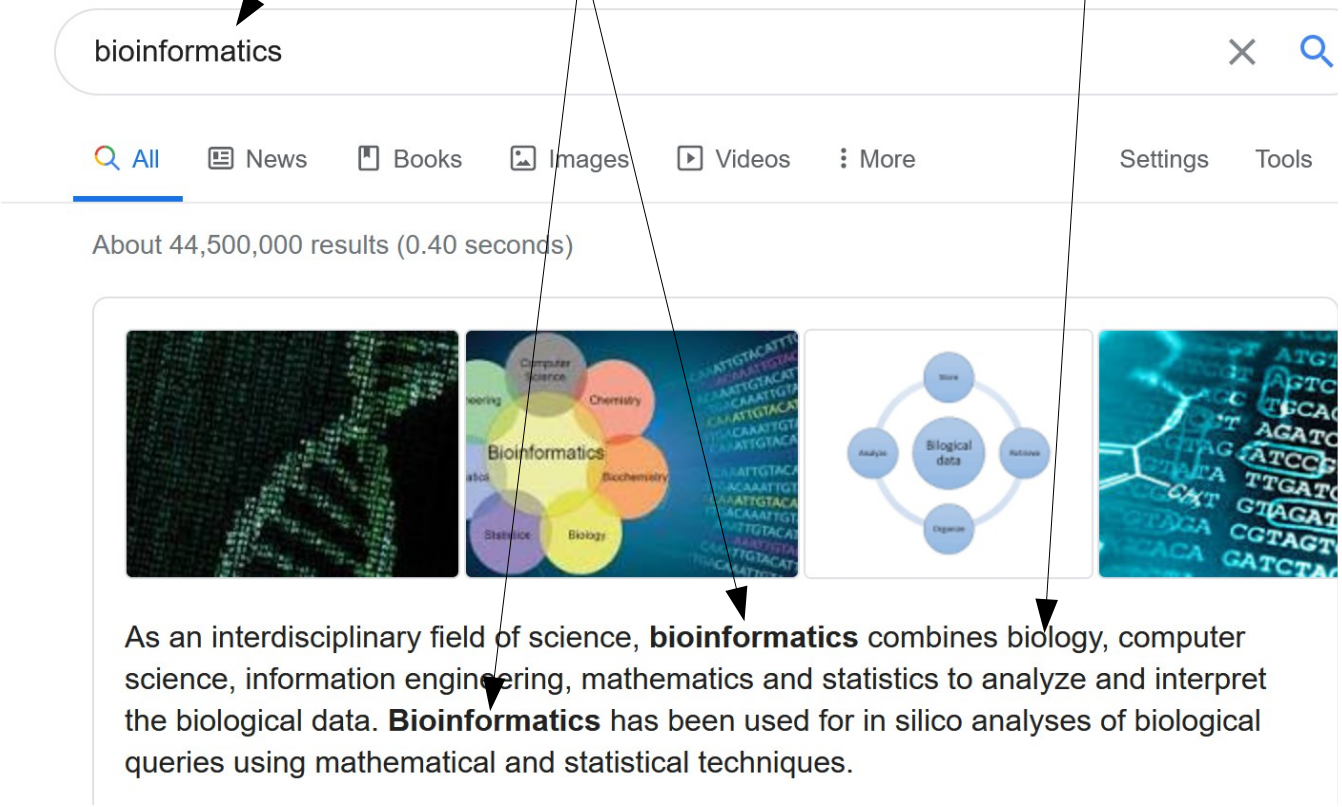
CMSC423: Bioinformatic Algorithms, Databases and Tools

Exact string matching:

Introduction

Exact string matching

- Find a word (**query or pattern**) within a **text**
- Think Google
- The output is a set of **matches** between the pattern and text



Stop and Think!

How would you write code to find all occurrences of a pattern within a text?

Stop and Think!

How would you solve the pattern matching problem if you were not allowed to use the string functions of your favorite programming language?

Sequence alignment: exact matching

```
ACAGGTACAGTTCCCTCGACACCTACTACCTAAG
CCTACT
 CCTACT
  CCTACT
   CCTACT
```

Text

Pattern

```
for i = 0 .. len(Text) {
  for j = 0 .. len(Pattern) {
     if (Pattern[j] != Text[i]) go to next i
  }
  if we got here pattern matches at i in Text
}
```

Stop and think?

What is the running time?

```
for i = 0 .. len(Text) {
  for j = 0 .. len(Pattern) {
     if (Pattern[j] != Text[i]) go to next i
  }
  if we got here pattern matches at i in Text
}
```

Answer

- Running time = O(m n)

  m = len(text); n = len(pattern)

- Stop and think: What pattern and text yield the worst-case runtime?
- Stop and think: What is the exact number of comparisons made? (i.e., not approximated by the O-notation)

NEXT: Can we do better?