No ID scan for attendance.

Survey at end of lecture.

You will be working in teams of 3-4 students from your lab. Please sit with students from your lab

Algorithms

- Approaches
- Growth
- Visualizing

What is an algorithm?

Algorithmic Approaches

- Exhaustive – try all possibilities, not usually the best approach, but helpful to understand the size of the solution space.
- Intelligent – There are various algorithm approaches that are “more efficient” than an exhaustive approach. You may already be aware of these intelligent algorithmic approaches, and use them in your daily life, without realizing it. The ability to use a particular intelligent algorithmic approach depends on the problem. Some examples of intelligent algorithmic approaches that you will learn about in later classes are greedy, divide and conquer, and dynamic programming.

How do we measure the “more efficient“ mentioned previously?

- In Computer Science we measure “more efficient” by the growth in work necessary to solve (i.e. runtime, memory usage) when the problem size grows.
- There are three cases to consider the growth of when applying an algorithm:
  - Best Case
  - Average Case
  - Worst Case

Example - Number Guessing Game

- I have a secret number between 1 and 100. You need to guess it. For each guess I will tell you if you are correct or too high or too low.
- What is an exhaustive algorithmic approach? What is an intelligent algorithmic approach?

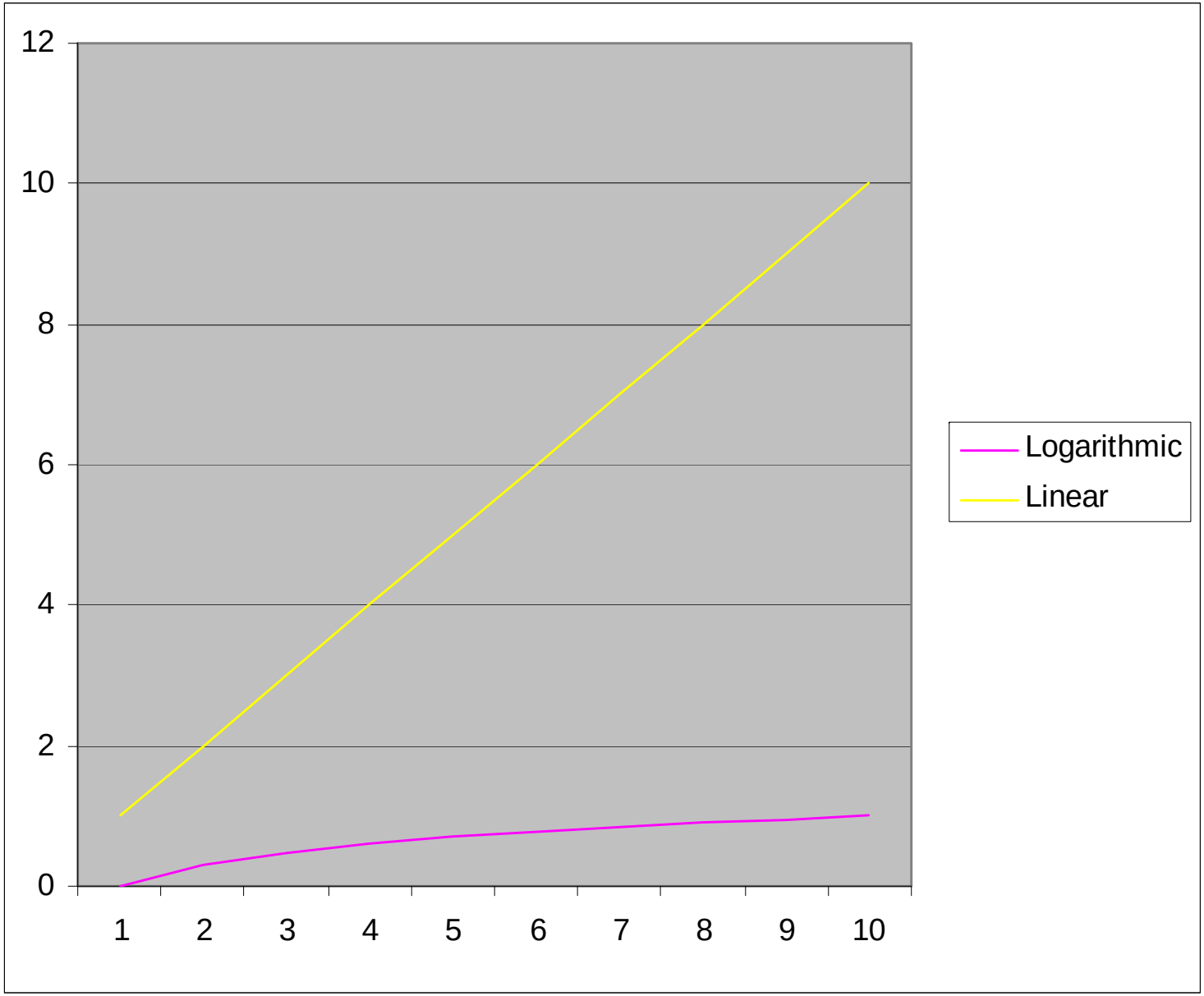
Measuring “more efficient“

- Recall we use growth in work necessary to solve when the problem size grows.
- Instead of a secret number from 1 to 100, what if I said the secret number was from 1 to 1000, or 1 to 10000, or 1 to 1000000?
- What is the work we measure?
- How would you compare the “more efficient“ of the exhaustive algorithmic approach vs. the intelligent algorithmic approach in best case, average case, worst case?

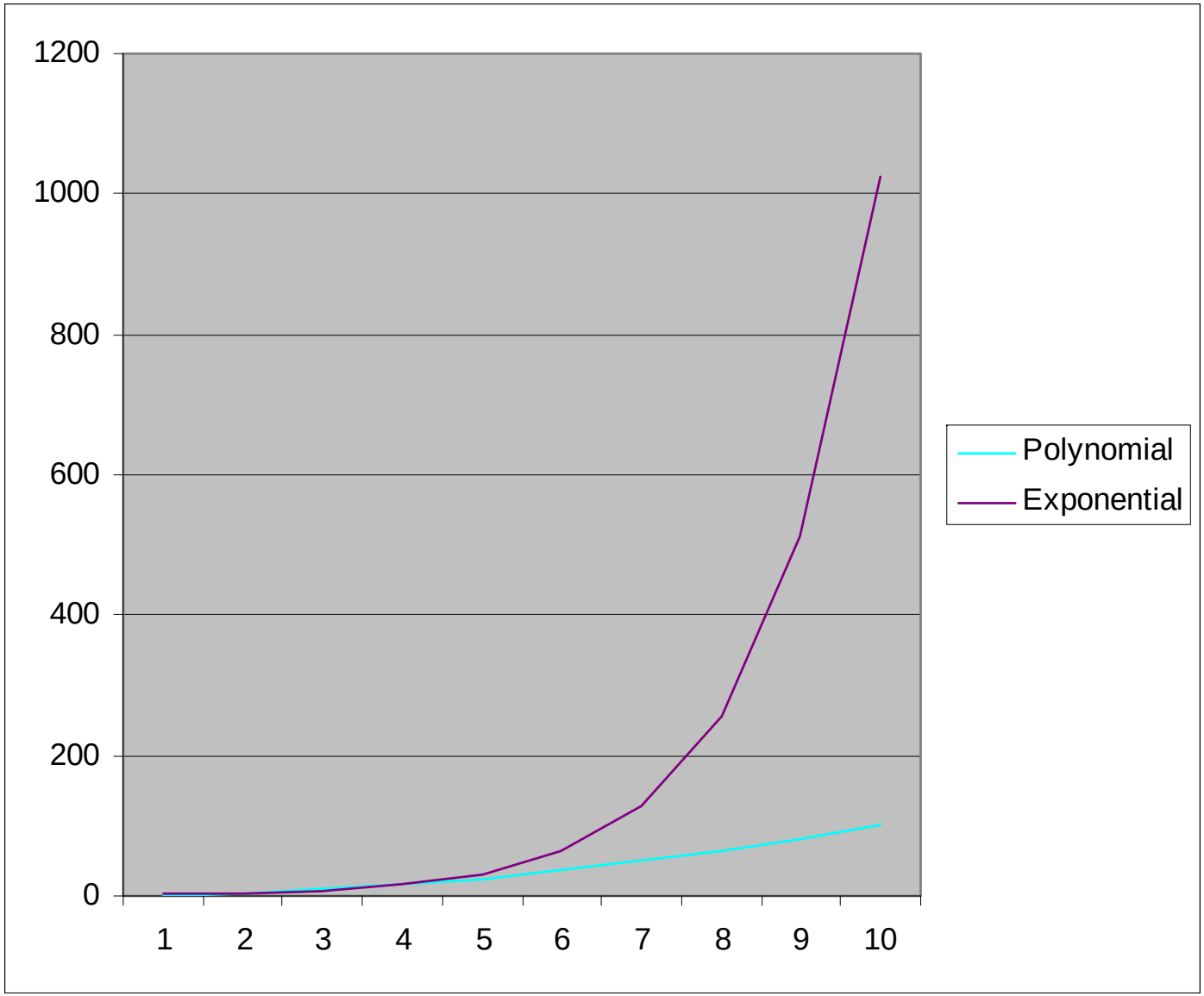
Example - Folding Challenge

- Try to fold on the dotted lines so the letters front and back spell the word COMPUTERS
- Strategies
  - Exhaustive
  - Intelligent
- What if the paper was 4 by 4 instead of 3 by 3, how much harder is the problem?

Growth



Growth


1200
1000
800
600
400
200
0
1
2
3
4
5
6
7
8
9
10
Polynomial
Exponential

Visualizing an Algorithm

- Sometimes visualizing an algorithm helps us understand how it works and makes it easier to remember the algorithm. It also may help us understand how much more work the algorithm needs to do if we increase the size of the problem.
- How can we visually represent solving the Number Guessing Game problem? And show how each algorithm (exhaustive or intelligent) executes on progressively larger and larger input?

Visualizing an Algorithm

- For each algorithm, make a prediction of what the visual representation of the Number Guessing Game problem will look like after the first 3 iterations. And then after the first 50 iterations.
- For each algorithm, estimate the total work necessary in the worst case to find a secret number out of a possible range of size “n”.

In Lecture Assignment - Sorting Algorithms – Visualization and Growth

- Complete the handout in 3-4 person teams, preferably with students from your lab.
- You will need the original to complete additional work in lab next week. If you worked with students in a different lab today, take a picture of each page to reference in lab next week.
- Complete the lecture survey. It is fine to talk with your team when completing.
- If you have a windows laptop, download and install IITSort from the CS100 www page before lab.