Computer Science Principles

CHAPTER 1 – COMPUTATIONAL THINKING

Announcements

Reminder: fill out course survey before end of today.

If you did not get an invite to the CampusWire discussion forum, please email me

Start reading Chapter 1 in the Conery textbook (Explorations in Computing)

Acknowledgement: These slides are revised versions of slides prepared by Prof. Arthur Lee, Tony Mione, and Pravin Pawar for earlier CSE 101 classes. Some slides are based on Prof. Kevin McDonald at SBU CSE 101 lecture notes and the textbook by John Conery.

What is Computer Science

- **Computer science** is all about using computers and computing technology to solve challenging, real-world problems in fields like science, medicine, business and society
- Computer science = computer programming ← Not True!
  - Computer programming is an important aspect of computer science
  - Computer programs often provide (parts of) the solutions to challenging technological problems
- Computer science is also not:
  - Computer literacy
  - Computer maintenance/repair

Computing systems

Let's take a tour of modern computing systems

A computing system consists of two major parts: the **hardware** and the **software**

Some hardware elements of a computer:

- Screen, keyboard, mouse
- Central processing unit (CPU), main memory
- Hard drives and other storage units

Types of software:

- Applications software, like office productivity programs, video games, web browsers
- Systems software, like operating systems, database systems

Computing systems

Can hardware exist without software?

- Sure, but is it useful?
  - General Purpose CPUs [Intel x86, core i5, i7, etc.] Not really
  - Specialized hardware [FPGAs, ASICs with functionality built into hardware] – Yes. However, their functionality was developed with software tools

Can software exist without hardware?

- In a literal sense, no – hardware is needed to execute software
- But, the problem-solving techniques used by programmers to create software do exist separately from the hardware and software

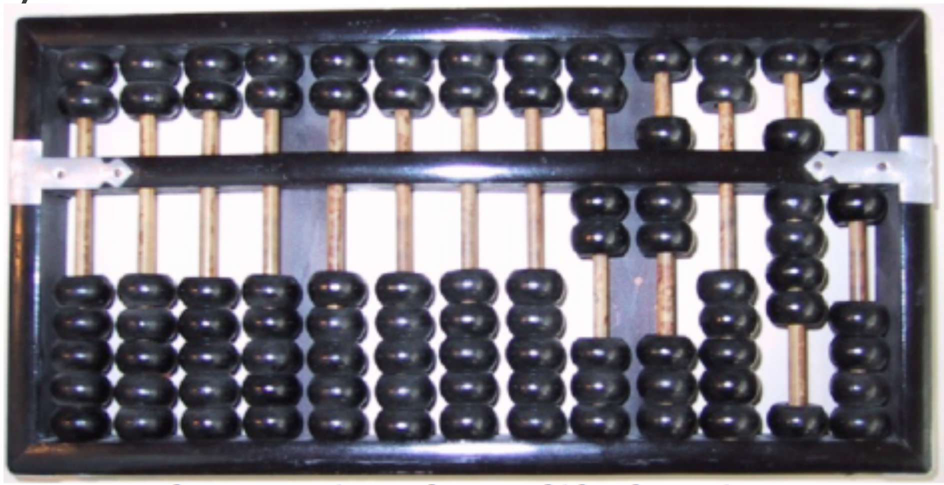
One more part to a computer system: data

- The software needs some kind of data to process: numbers, text, images, sound, video

Quick History of Computing

A quick history of computing

- We think of computers as modern inventions, but computing devices go back thousands of years and have many of the same basic features of modern digital computers.
- **Abacus** – an early device to record numeric values and do basic arithmetic (16th century B.C.)



What does an abacus have to do with laptops, smartphones and tablet computers???

A quick history of computing

Modern computers borrow four important concepts from the abacus:

1. Storage
2. Data Representation
3. Calculation
4. User Interface

A quick history of computing

1. Storage
   - An abacus stores numbers, which are the most fundamental type of data in modern computing.
   - In a modern computer, all data – text, images, audio, video – is represented using binary numbers (1s and 0s)

2. Data Representation
   - The abacus represents numbers using beads on spindles.
   - Modern computers employ a variety of techniques for representing data on storage media:
     - Magnetic – e.g. used with hard disk drives (HDD)
     - Optical – e.g. used with CDs, DVDs
     - Electrical – e.g. used with solid-state drives (SSD)

A quick history of computing

3. Calculation
   - By moving beads on abacus spindles, user can perform addition, subtraction, multiplication, and division
   - Modern computers contain powerful central processing units that perform calculations at astonishing speeds

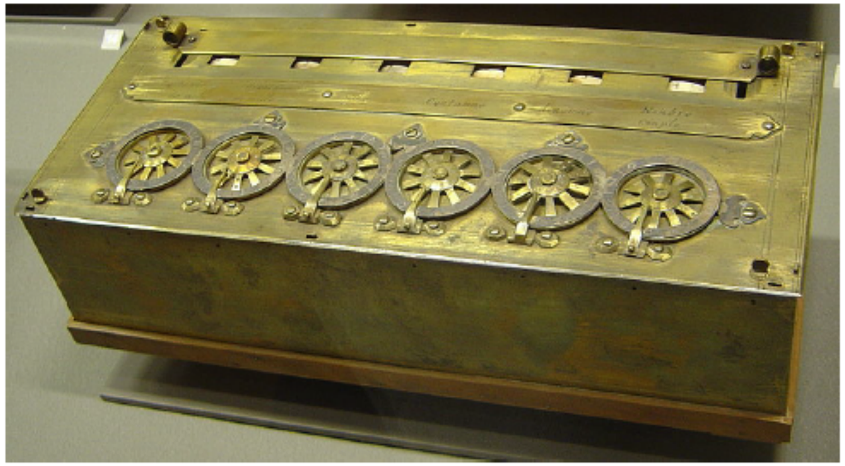
4. User Interface
   - The beads and spindles on the abacus
   - Modern computers provide a wide variety of input and output devices for the user

A quick history of computing

In the 17th century people began tinkering with physical devices that could do computations and calculations

Blaise Pascal

- French mathematician and philosopher
- Designed and built a mechanical calculator

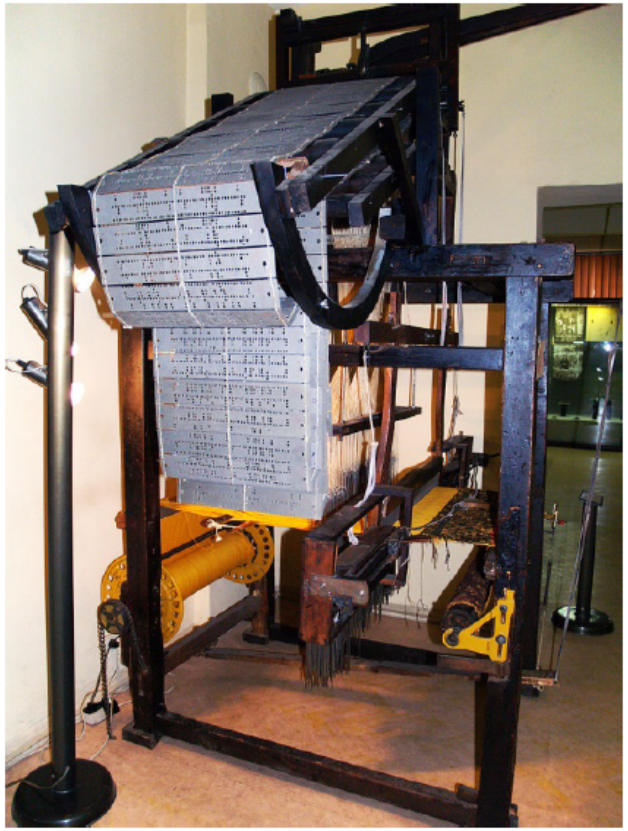


Calculator could only do addition and subtraction

- Input is given using dials
- Output is read on small windows above each dial

Programmable devices

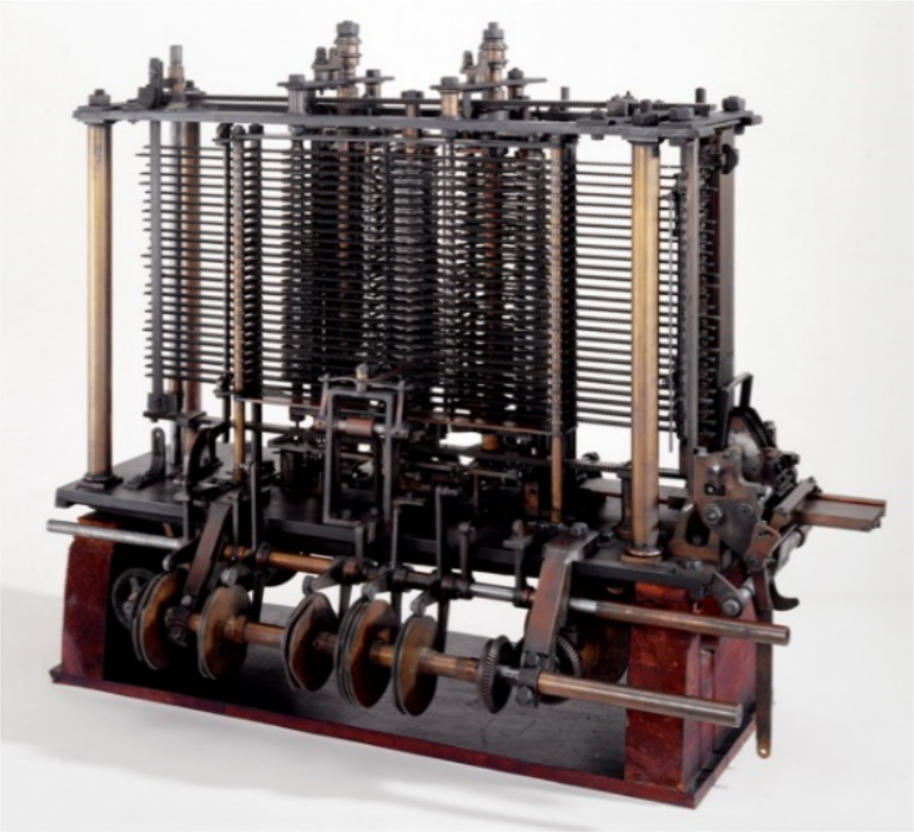
- Pascal's calculator and other similar devices of that time were not programmable
- One of the first programmable devices in history was a loom
- Joseph Marie Jacquard's loom (1804) could be programmed by feeding in a set of punched cards



Programmable devices

Another leap forward came in the 19th century with Charles Babbage's design of the Analytical Engine – a mechanical, programmable computer

- It was never actually built in Babbage's time due to a lack of manufacturing capabilities
- Design called for punched cards to be fed into the machine to program it to perform mathematical calculations
- Output would go to a printer or more punched cards



Programmable computers

Now, move forward to the 20th and 21st centuries

A modern computer is defined by three basic requirements

1. Must be electronic and not exclusively mechanical.

2. Must be digital, not analog

   Digital devices use discrete values (digits), not a continuous range of values to represent data. (i.e. digital vs mercury-based thermometer)

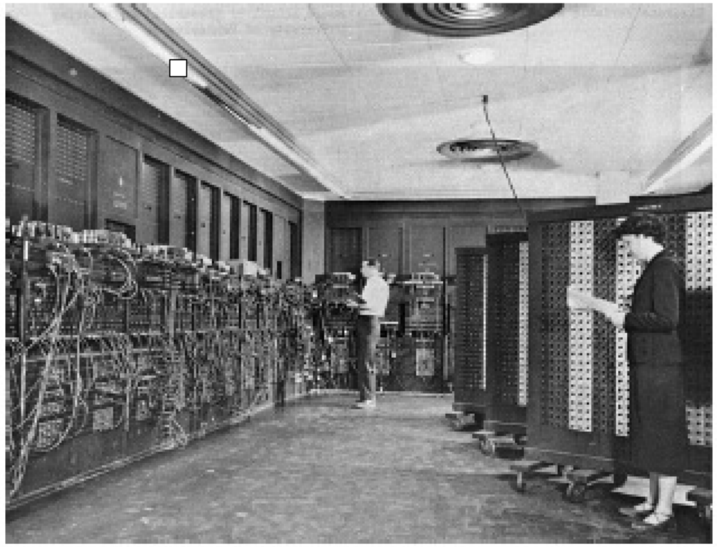
3. Must employ the **stored-program concept**

   The device can be reprogrammed by changing the instructions stored in the memory of the computer

Programmable computers

ENIAC (Electronic Numerical Integrator and Computer)

- Built in the 1940s
- Among the first computers to employ the stored-program concept



Modern computers have four major components:

- Input device(s)
- Output device(s)
- Memory – for data storage, both temporary & permanent
- Processor – for doing computations

Programmable computers

Again, the **stored-program concept** is the idea that programs (software) along with their data are *stored* (saved) in the memory of a computer

- Not referring to storage on hard drives, flash drives or CDs
- Referring to **main memory** of the computer, sometimes called the **RAM** (random access memory)

A modern processor

- Reads the **machine instructions** *stored* as 1s and 0s in the main memory
- Executes those instructions in sequence
- Key point: these instructions can be changed to reprogram the computer to do new tasks

Transistors

A variety of devices have been used to represent digits and to control the operation of computing machines

In the 1940s:

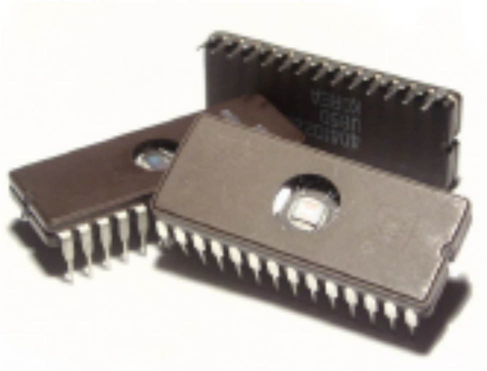
- Bardeen, Brattain, and Shockley invented the transistor, which is an electronic switch with no moving parts

In the 1950s and 1960s:

- Kilby, Noyce, and others used transistors to develop integrated circuits
- Devised a way to manufacture thousands – later, millions and billions – of transistors on a single wafer of silicon

A single chip contains:

- An integrated circuit
- A ceramic or plastic case
- External pins to attach it to a circuit board



Transistors

Noyce and businessman Gordon Moore commercialized this technology by co-founding Intel Corporation in 1968

Manufacturing technologies improved in the 1950s and 1960s:

- Engineers were able to pack many more transistors per unit area on silicon wafers
- **Moore's law**: Moore observed that the number of transistors within an integrated circuit was doubling about every 2 years.
    - The trend continued for around 40 years.
    - But transistors can be only so small!

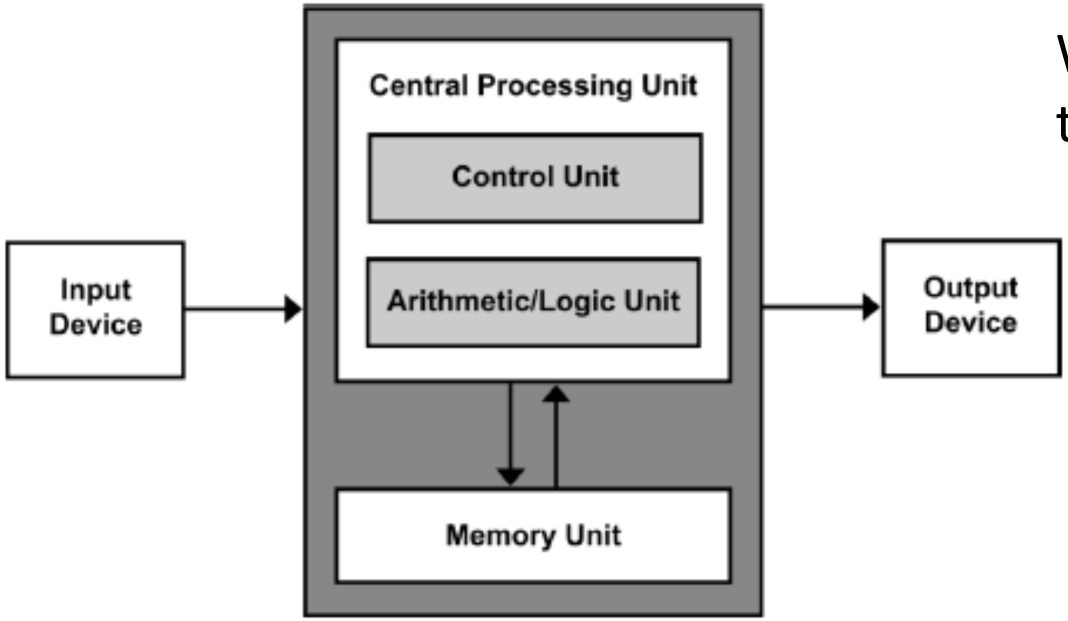
Combating miniaturization challenges:

- Intel, AMD (Advanced Micro Devices) and others now make processors that feature multiple processing cores that perform calculations in parallel with each other

Modern computer architecture

The stored program approach used today is implemented using von Neumann architecture, named after U.S. mathematician John von Neumann

This architecture contains input devices, output devices, a processor and a memory unit



Will now look at how they work together to form a functioning computer

Modern computer architecture

In modern computers (PCs), the major components in a von Neumann machine reside physically in a circuit board called the **motherboard**

- The CPU, memory, expansion cards and other components are plugged into slots so they can be replaced
- Hard drives, CD drives, and other storage devices are connected to the motherboard through cables

The central processing unit (CPU) is the “brain” of the machine

- Its **arithmetic/logic unit** (ALU) performs millions or billions of calculations per second
- The **control unit** is the main organizing force of the computer and directs the operation of the ALU

Modern computer architecture

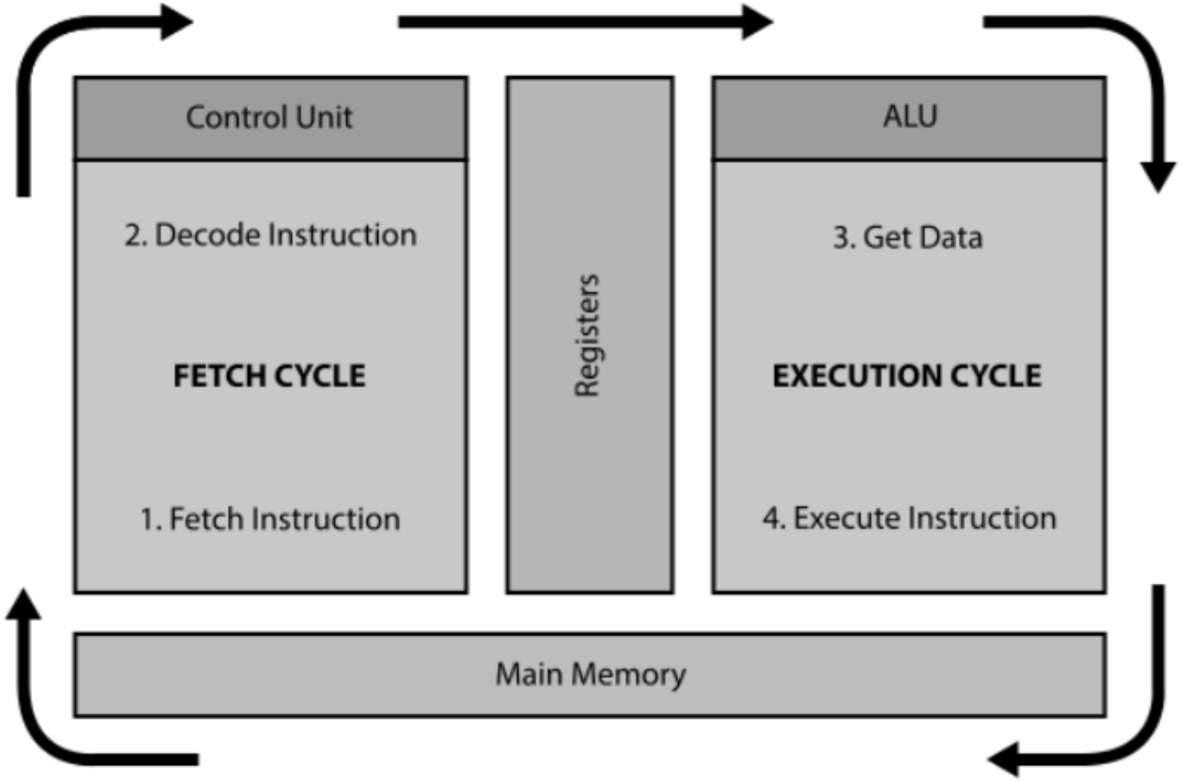
The fetch-decode-execute cycle

The system sends electrical signals that encode machine instructions and data

- The CPU **fetches** the instructions and data from memory as needed
- The control unit **decodes** each instruction to figure out what it is (an addition operation, subtraction, etc.)
- Data values (e.g., numbers to be added and their resultant sum) are stored temporarily in memory cells called **registers** within the CPU
- The ALU **executes** the instruction, saving the result in the registers and main memory

This whole process is known as the **fetch-decode-execute cycle**

The fetch-decode-execute cycle



What about the software?

Software consists of instructions for the CPU to execute

- CPUs “understand” something called **machine language**, which consists of 0s and 1s
- A single instruction for a modern computer might consist of some combination of 32 or 64 0s and 1s

Most programming done today uses **high-level programming languages**, which consist of English and English-like words and some mathematical notation

We will learn the basics of Python, a popular, easy-to-learn, high-level programming language

Computational Thinking

What is computational thinking?

➔ How computer scientists think – how they reason and work through problems

Computer science encompasses many sub-disciplines that support the goal of solving problems:

- Computer theory areas ➔ these are the heart and soul of computer science
  - Algorithms
  - Data structures
- Computer systems areas
  - Hardware design
  - Operating systems
  - Networks
- Computer software and applications
  - Software engineering
  - Programming languages
  - Databases
  - Artificial intelligence
  - Computer graphics

A major goal of this course is to help you develop your computational thinking and problem solving skills

A modern computing problem

Electronic health records are very important

Consider issues (technical + others) that arise providing a system to medical professionals and others who need access to digital medical records:

- What data will be stored? How? In what format?
- How will the data be accessed and displayed?
- Who will have access? How will the data be secured?
- How will the data be backed up and preserved?

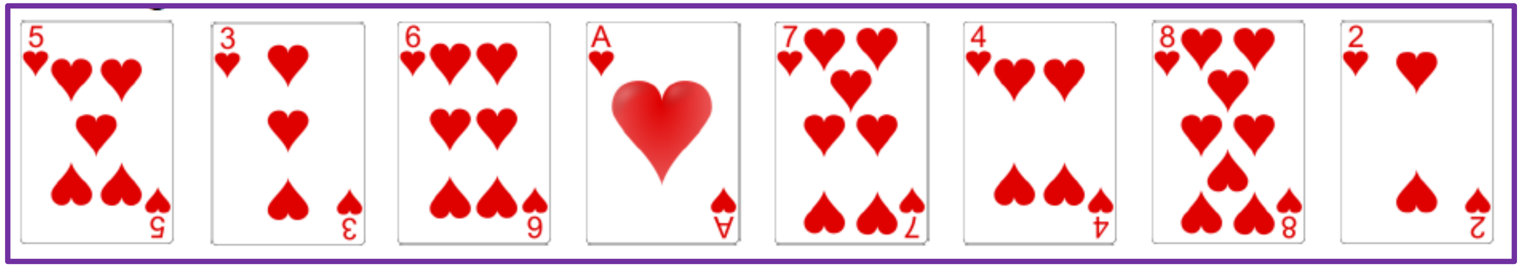
Answering these questions requires computational thinking

A classic problem: Sorting data

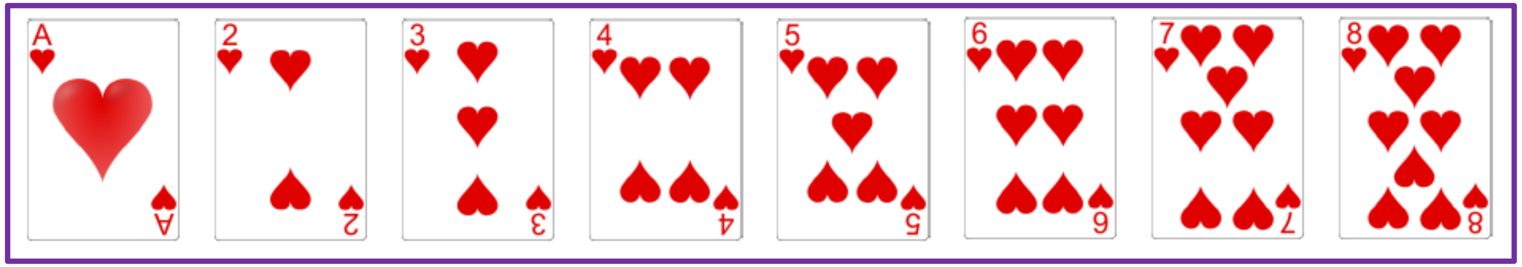
Sorting: an important problem – arises frequently in computer science

- Suppose there is a deck of cards to be put in order
- Example: We have the Ace up to 8 of Hearts

Given:



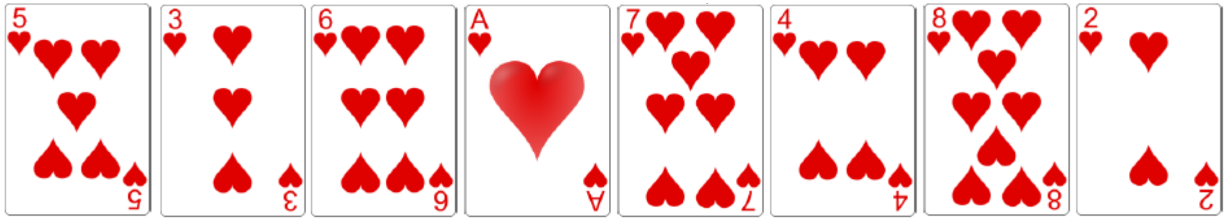
Want the following:



A classic problem: Sorting data

You want to explain to a young child how to put the cards in order

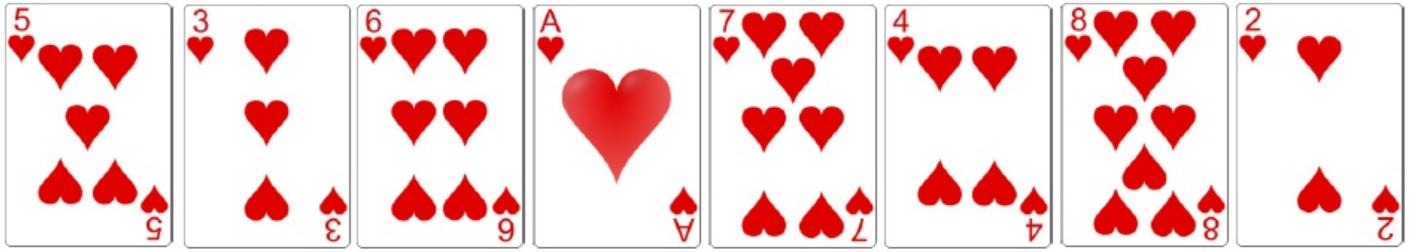
- What steps would you give?



A classic problem: Sorting data

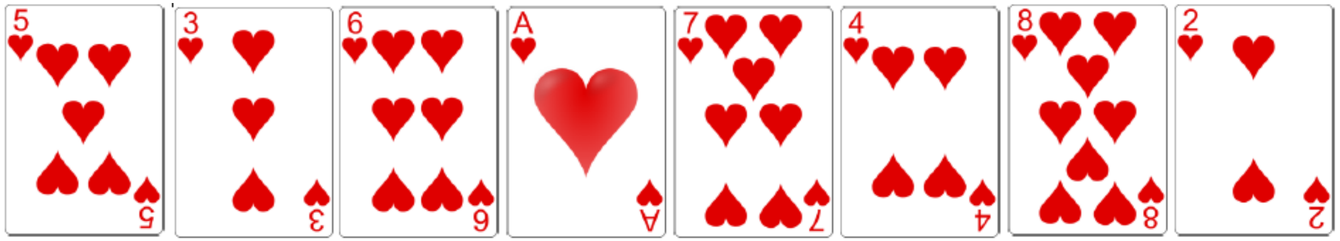
One sorting technique is called selection sort

- It repeatedly searches for and swaps cards in the list



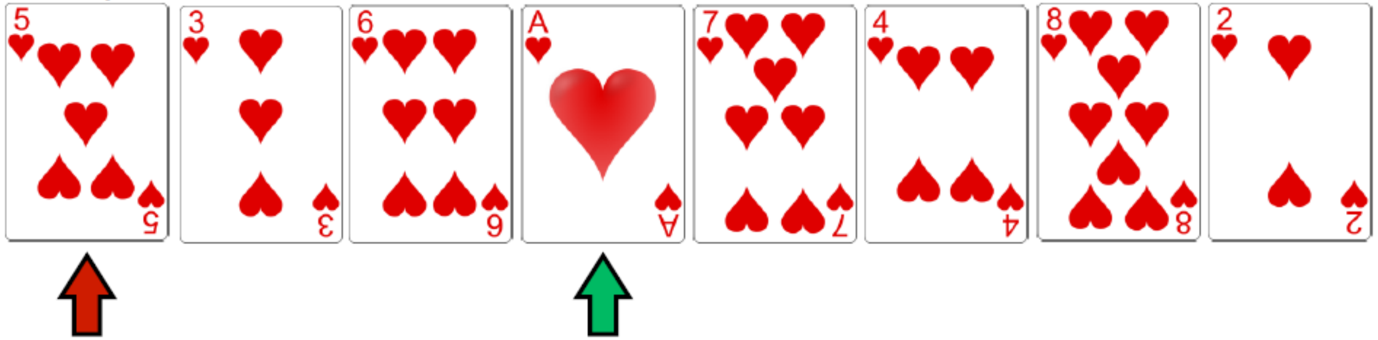
Selection sort

First, find the smallest item and exchange it with the card in the first position



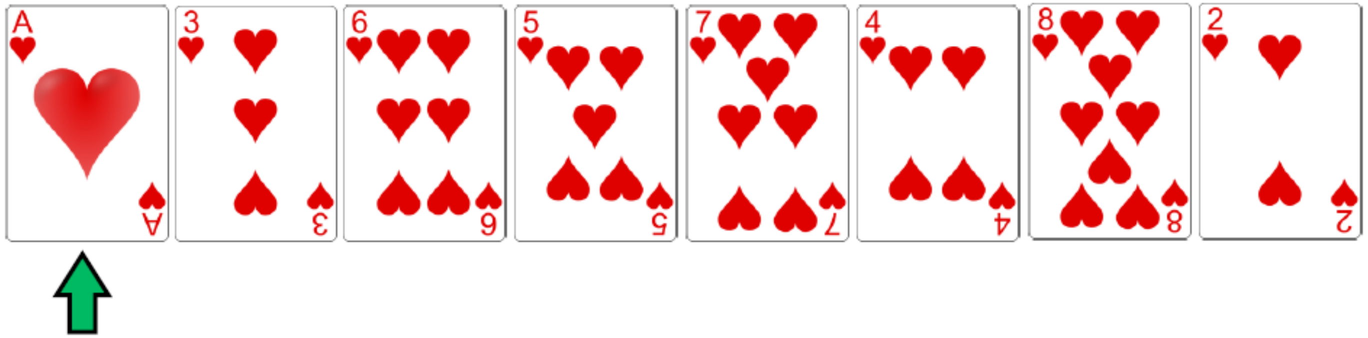
Selection sort

Select the smallest item and exchange it with the card in the first position



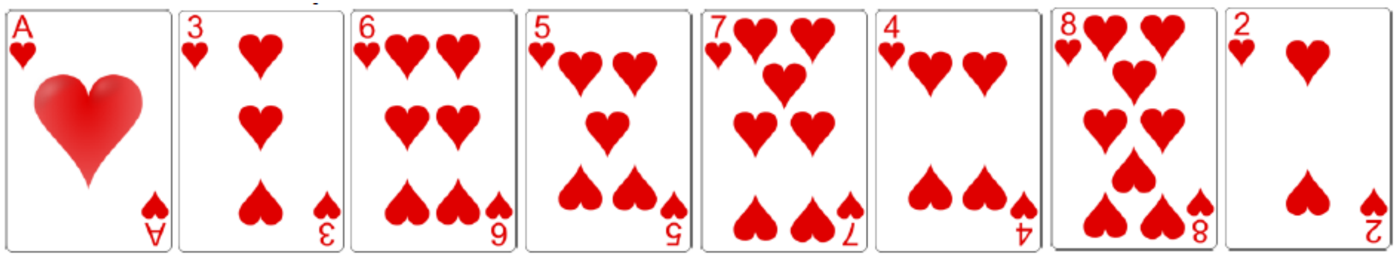
Selection sort

How the cards appear after the exchange:



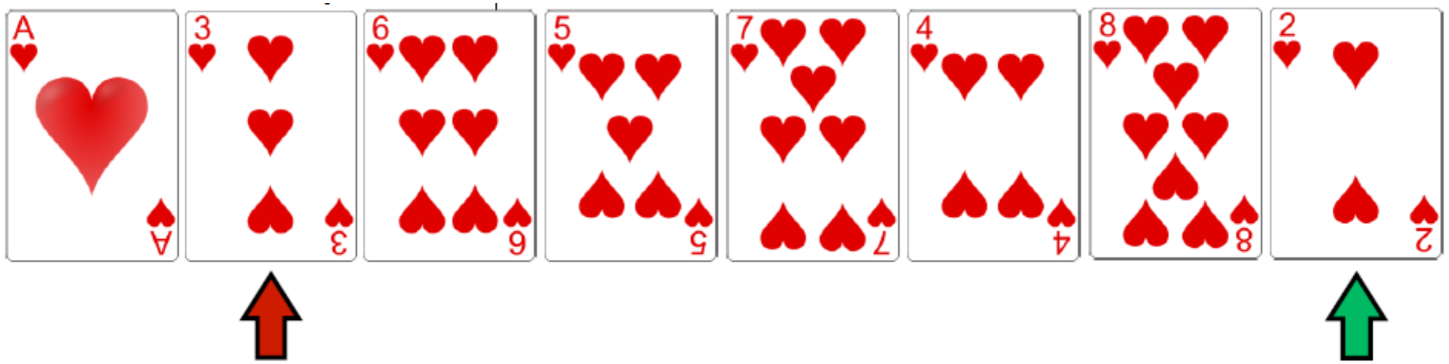
Selection sort

Next, **select** the second-smallest item and exchange it with the card in the 2nd position



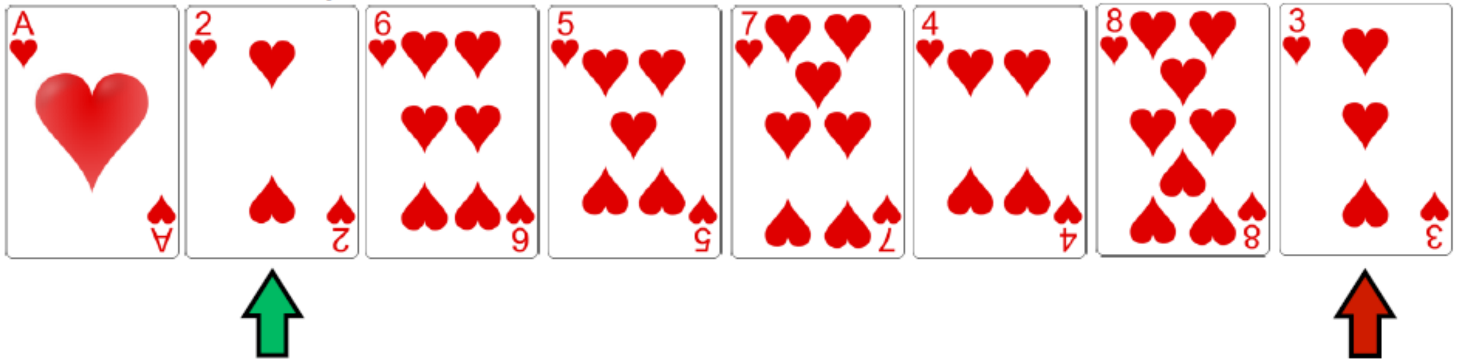
Selection sort

Select the second-smallest item and exchange it with the card in the 2nd position



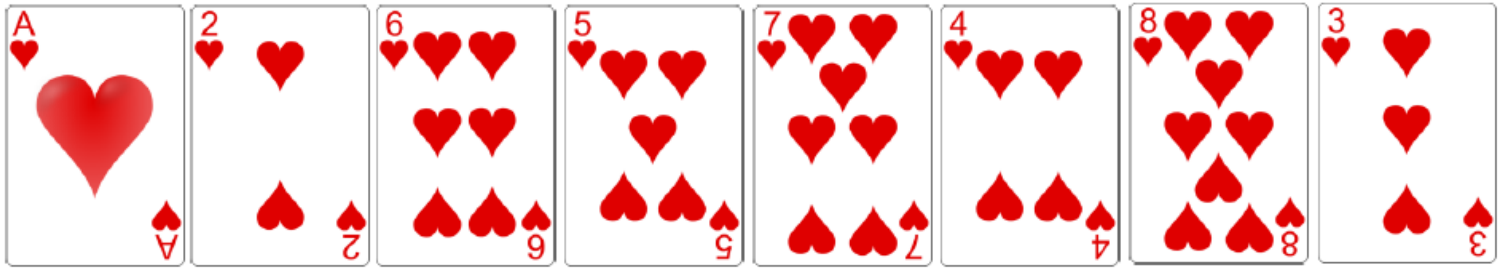
Selection sort

How the cards appear after the exchange:



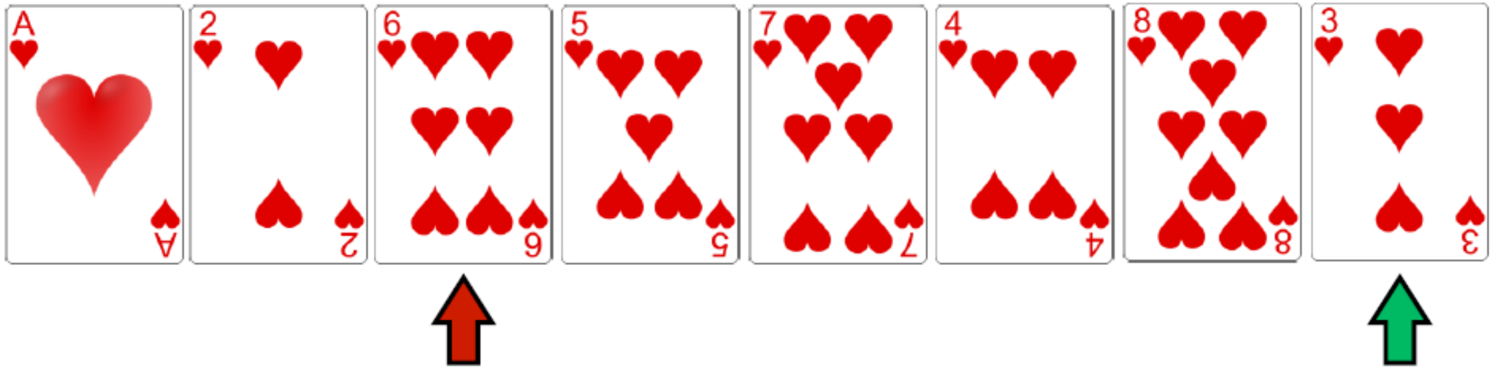
Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted



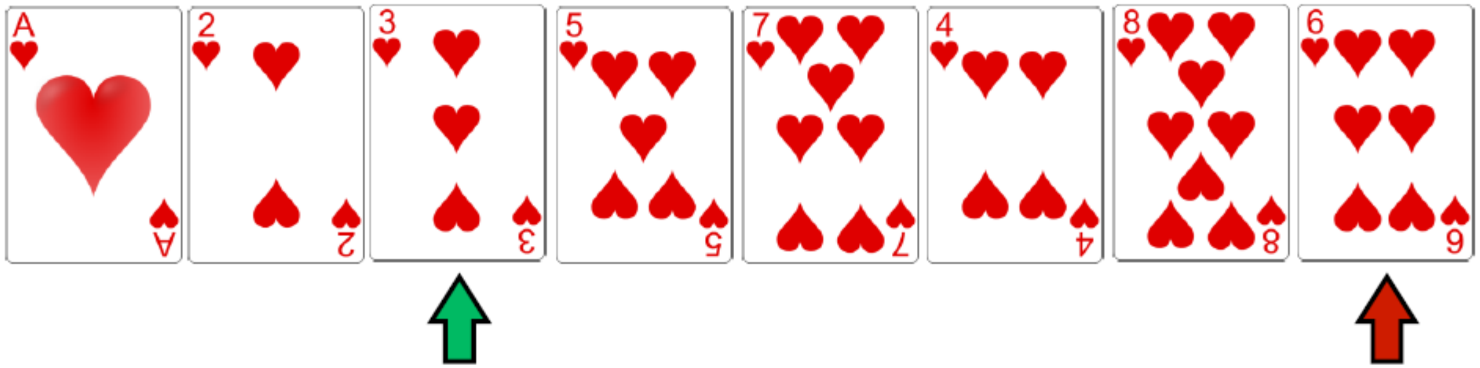
Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted



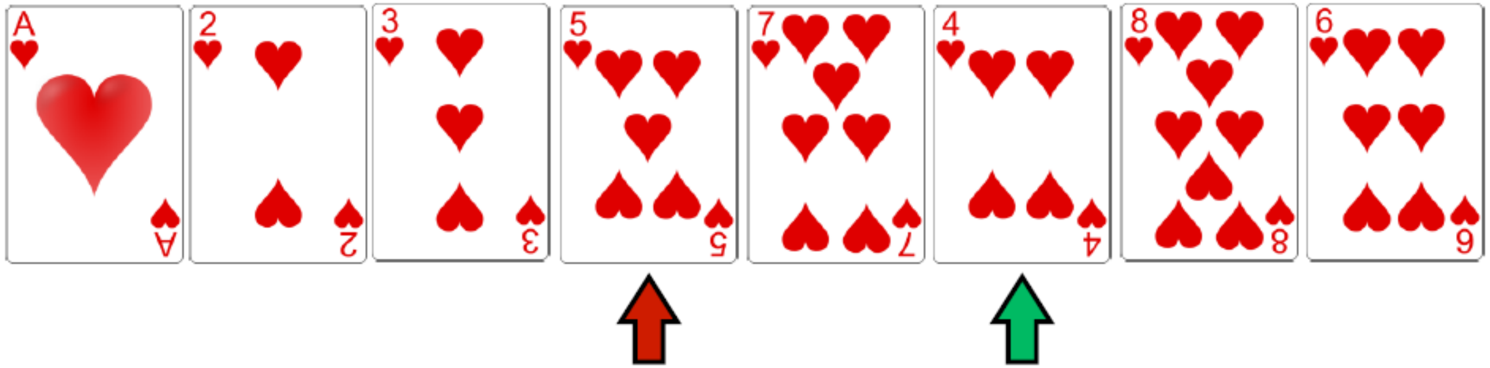
Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted



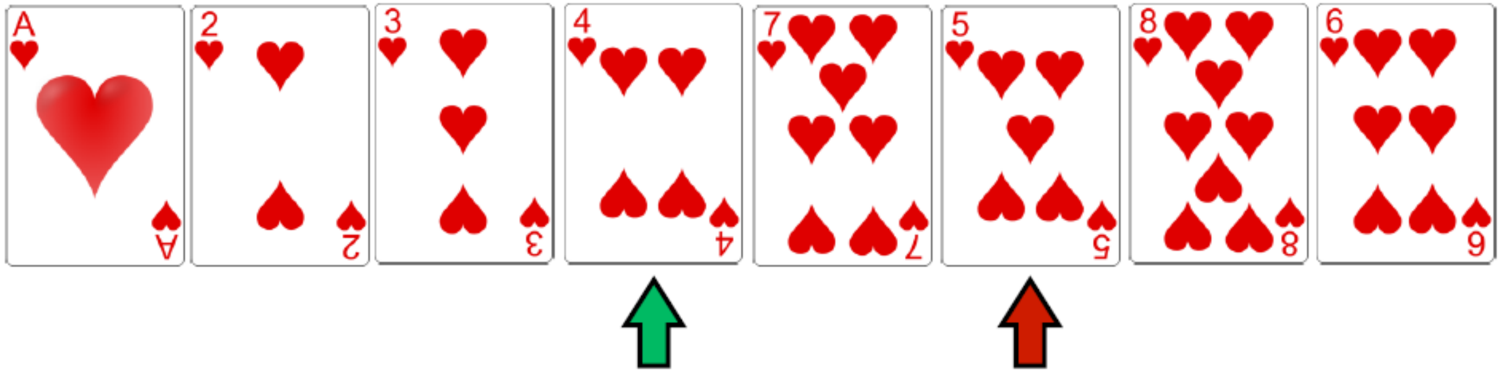
Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted



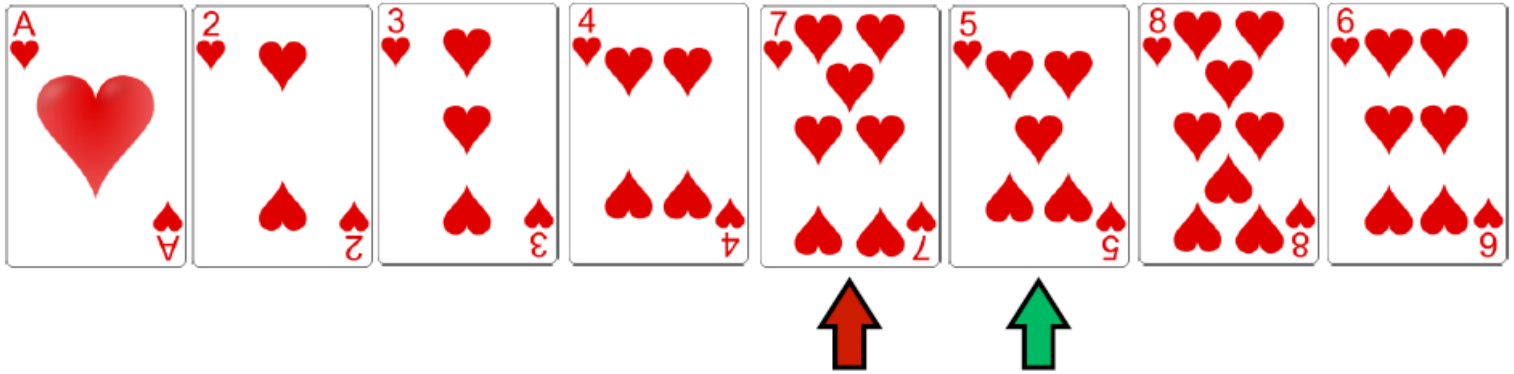
Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted



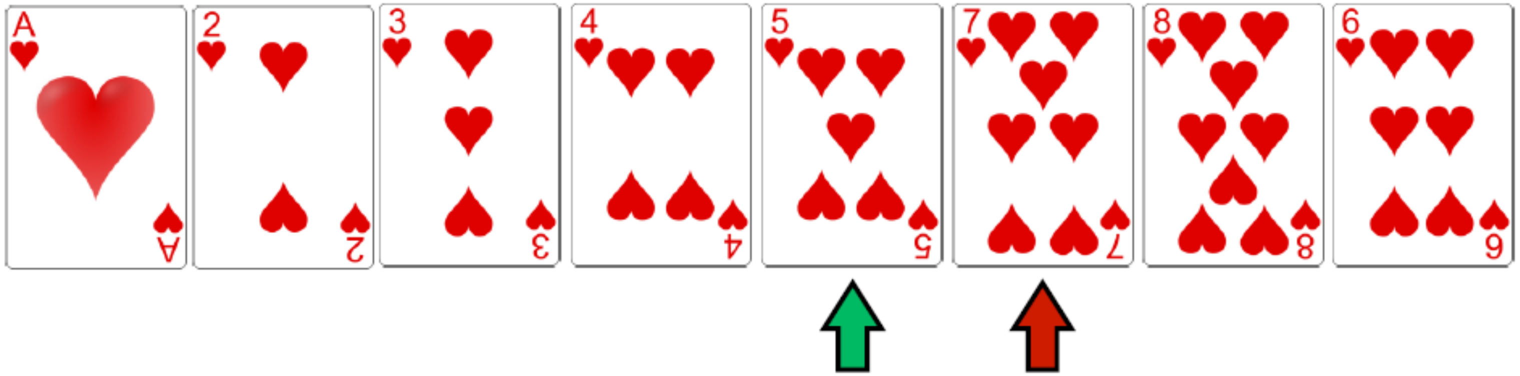
Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted



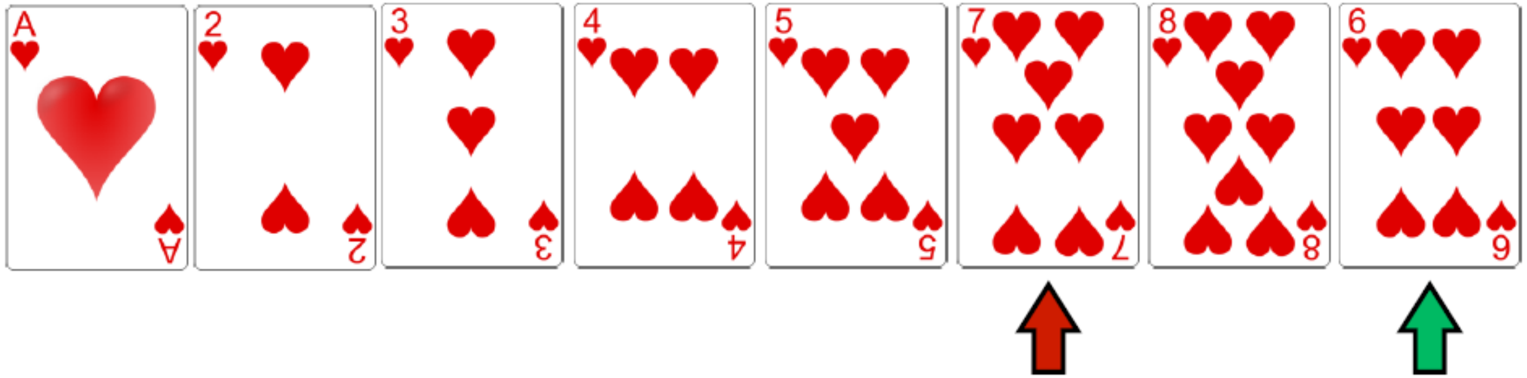
Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted



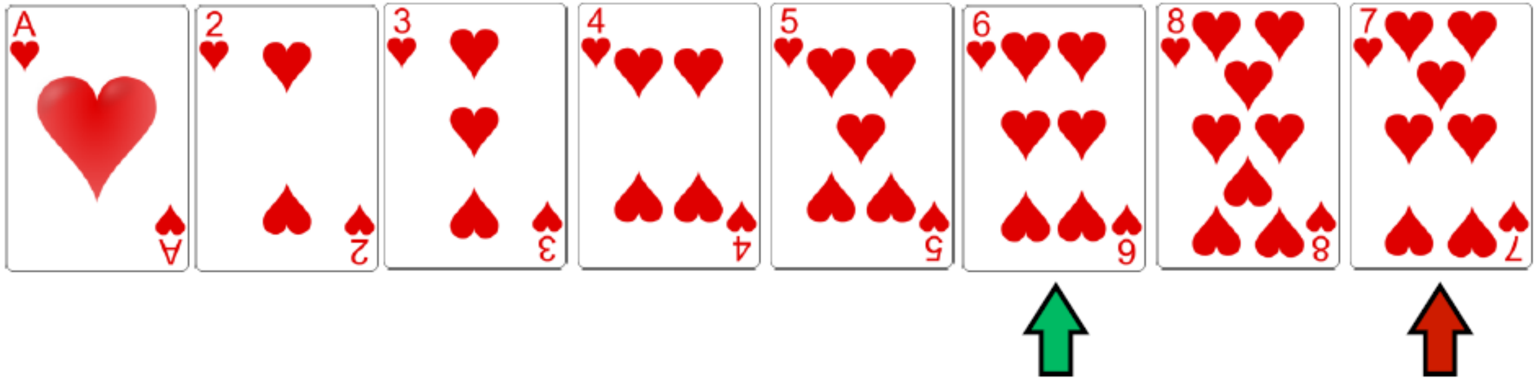
Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted



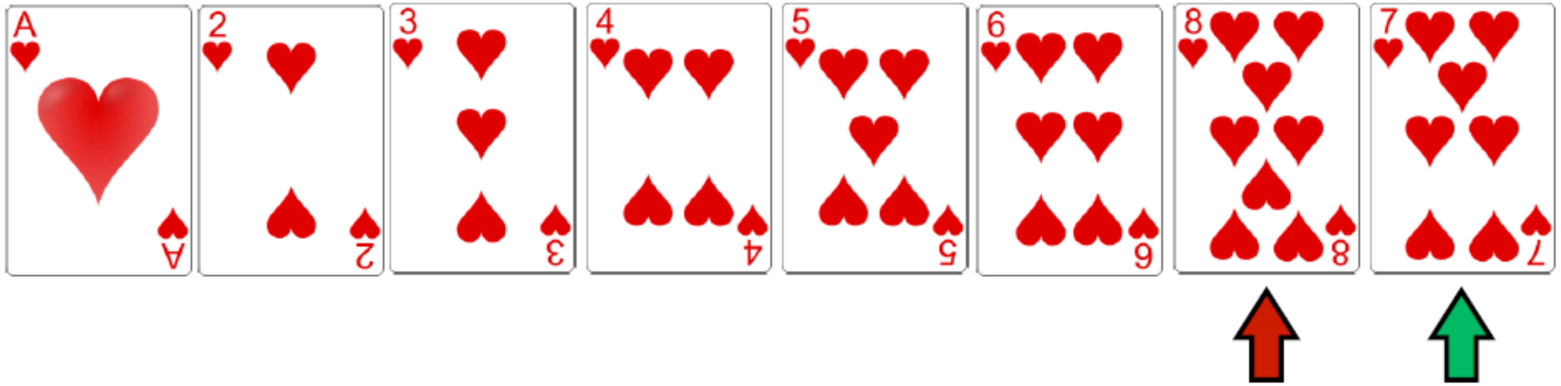
Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted



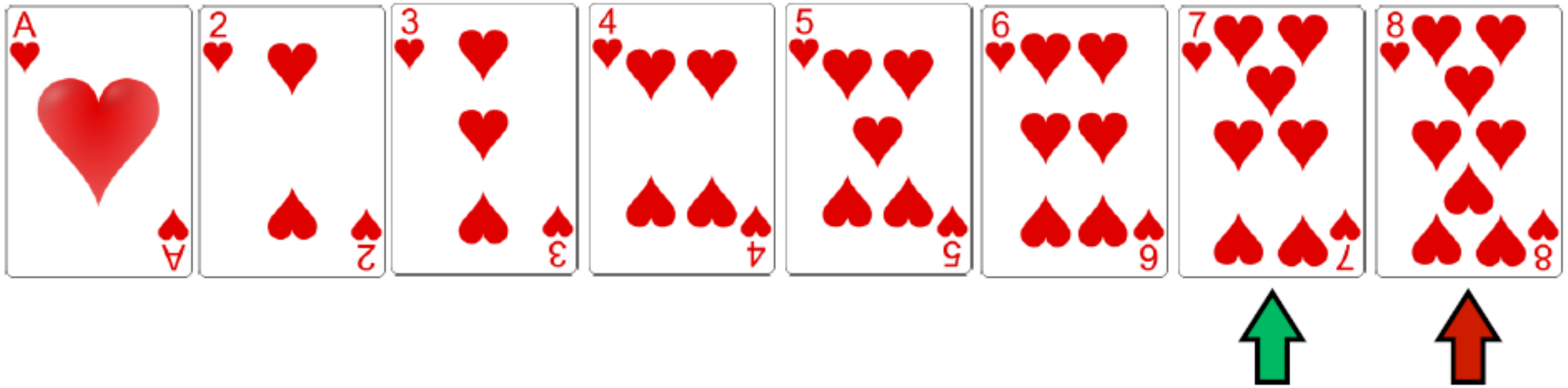
Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted



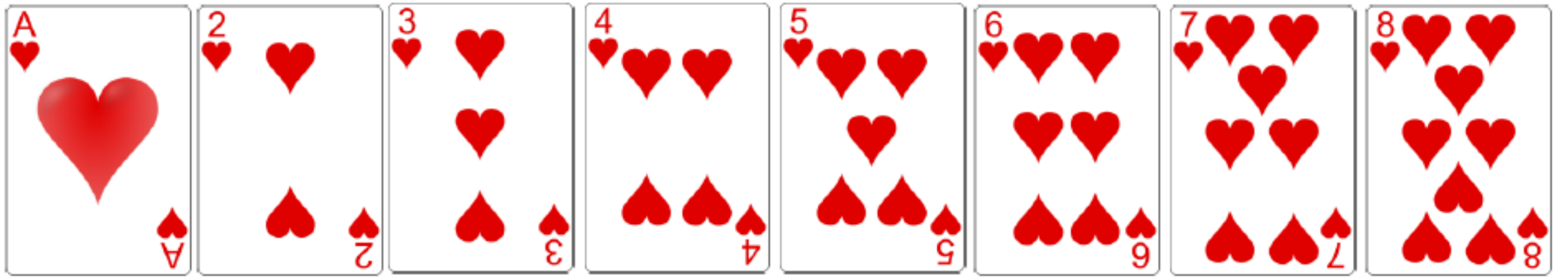
Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted



Selection sort

Continue in this fashion, **selecting** the third-smallest, fourth-smallest, etc., until the cards are sorted

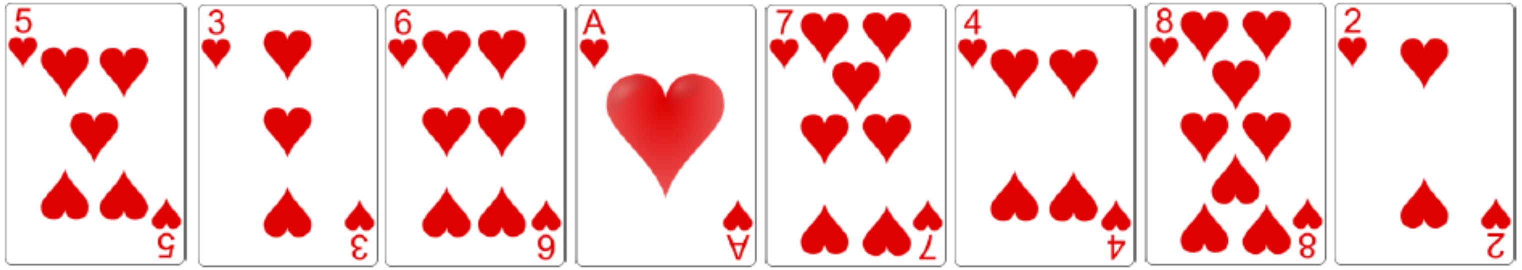


Finished!

Different Technique: Insertion Sort

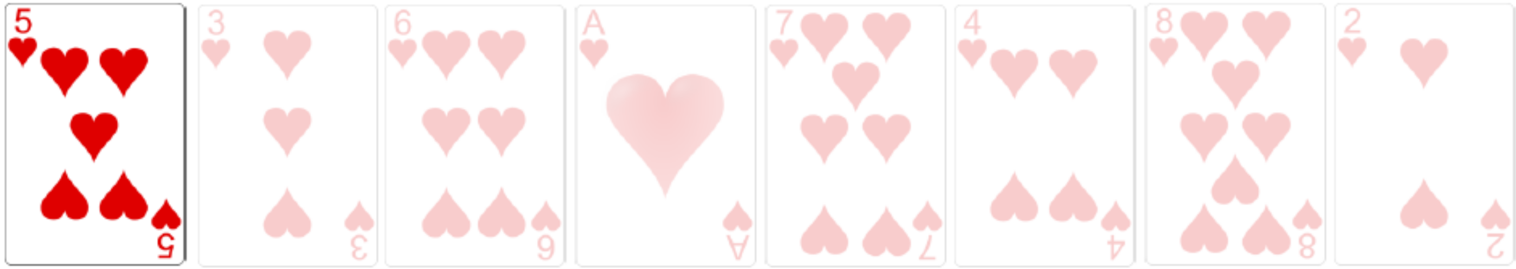
Another sorting technique is insertion sort

- It repeatedly inserts the “next” card into its correct spot



Insertion sort

Begin by leaving the first card (#5) where it is

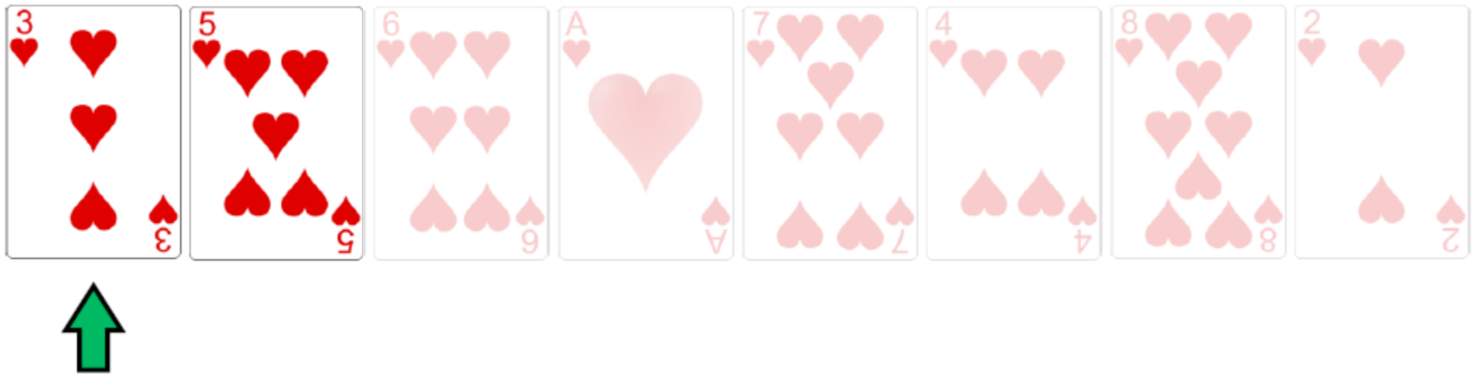


Insertion sort

- The second card (#3) is smaller than the first card
- Insert it in front of the first card

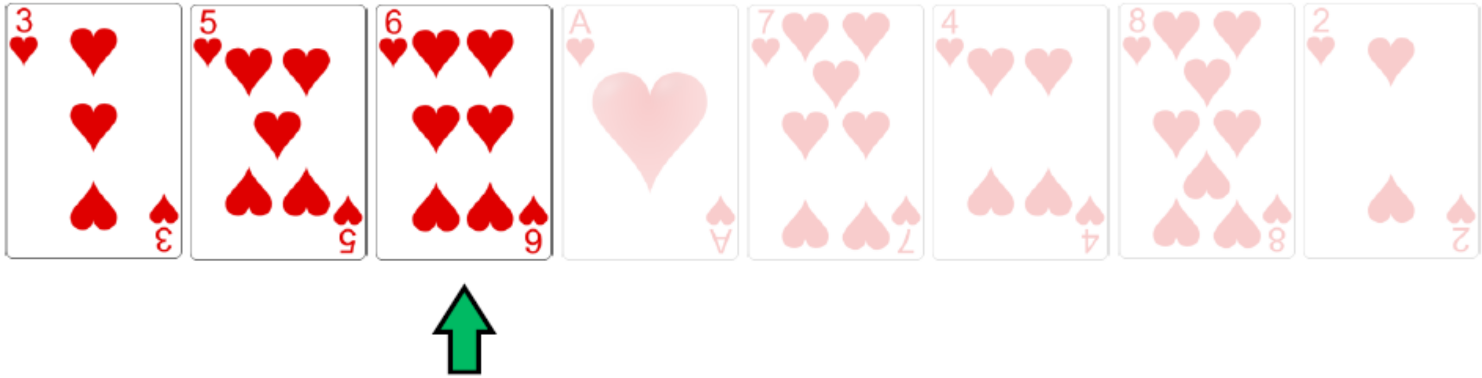
Insertion sort

- The second card (#3) is smaller than the first card
- Insert it in front of the first card



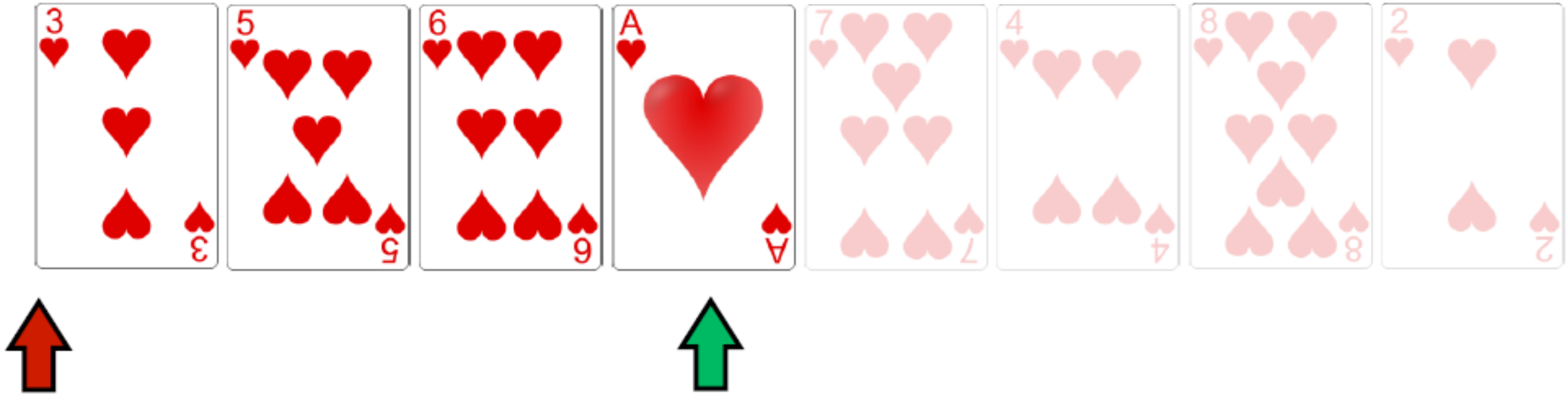
Insertion sort

- The third card (#6) is larger than the first two cards
- So, it does not need to move



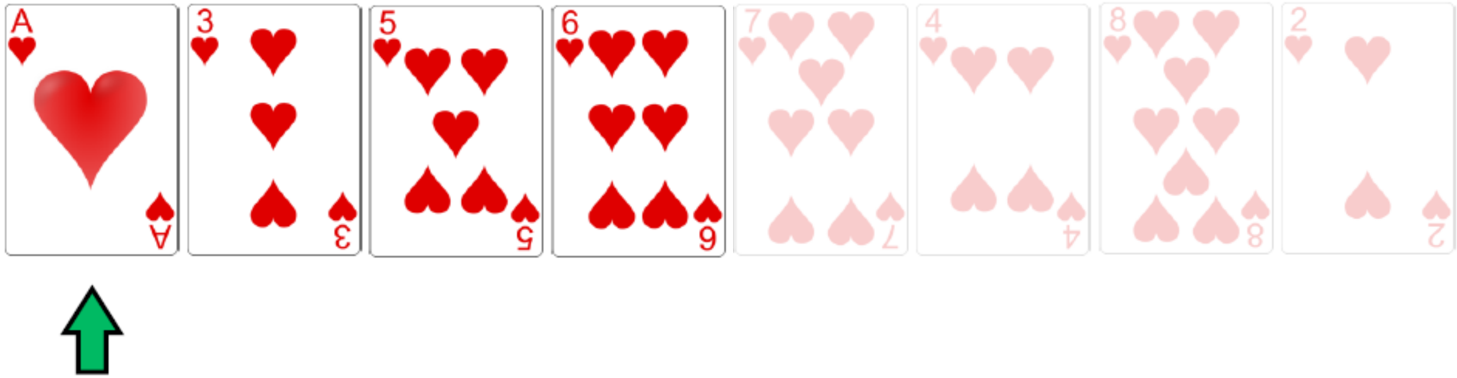
Insertion sort

- The fourth card (#1) is smaller than the first three cards
- Insert it in front of the first card, shifting the others



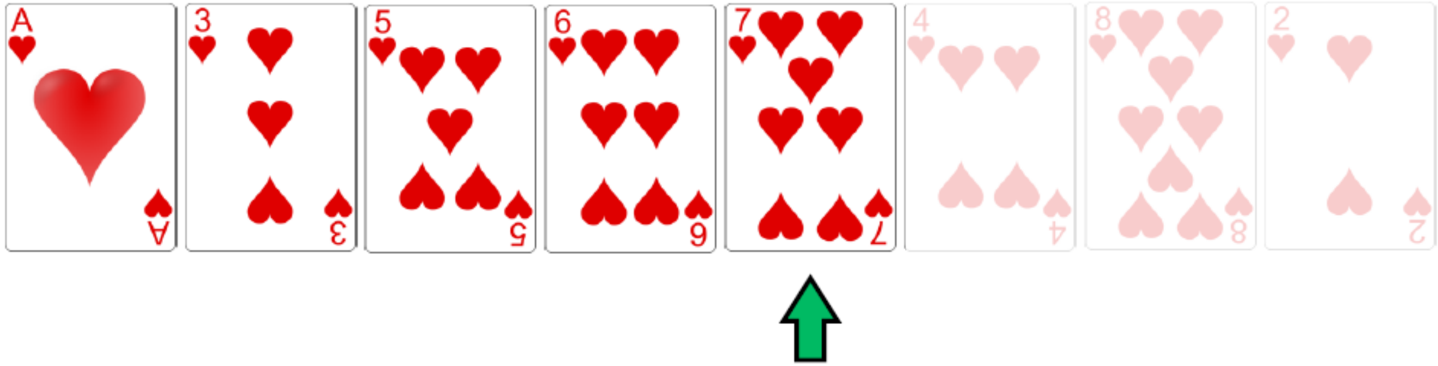
Insertion sort

- The fourth card (#1) is smaller than the first three cards
- Insert it in front of the first card, shifting the others



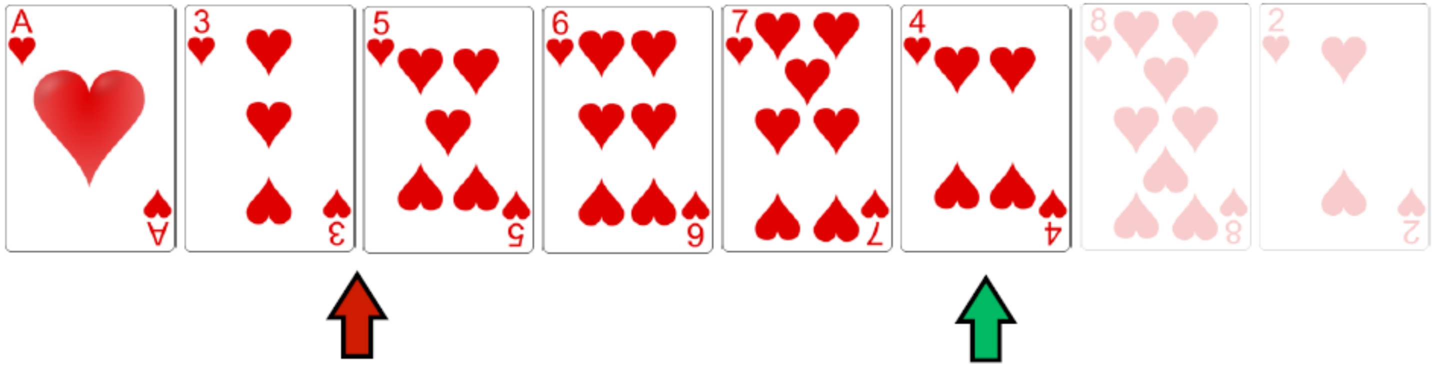
Insertion sort

- The fifth card (#7) is larger than the first four cards
- So, it does not need to move



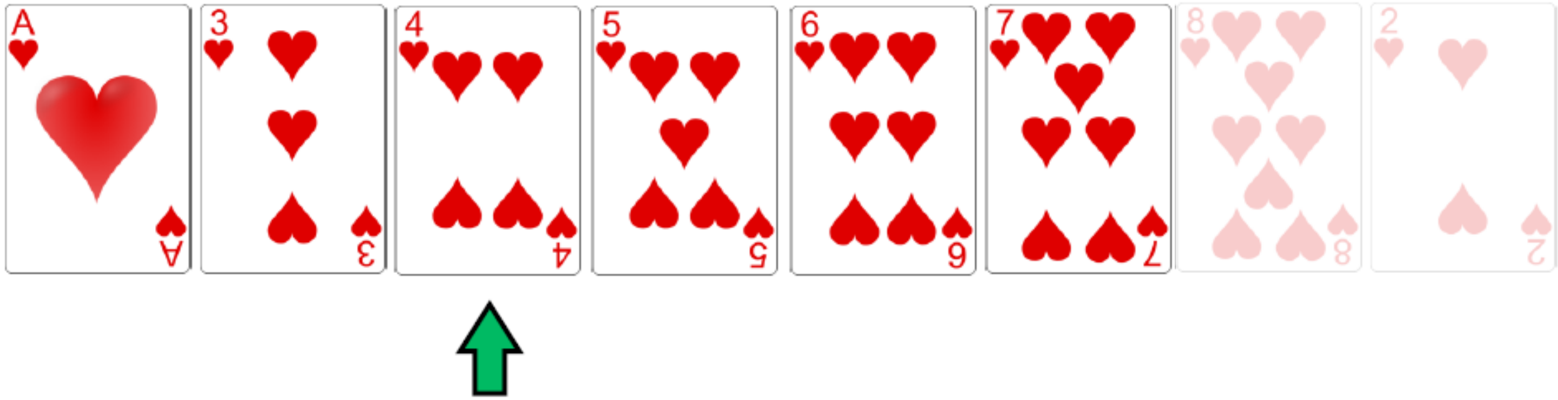
Insertion sort

- The sixth card (#4) should be inserted in between the second (#3) and third (#5) cards



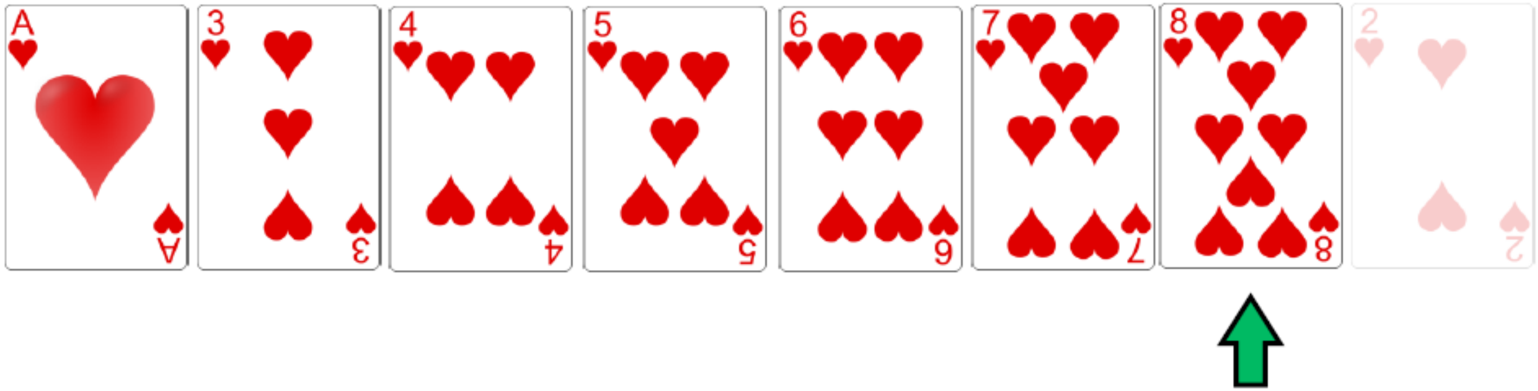
Insertion sort

- The sixth card (#4) should be inserted in between the second (#3) and third (#5) cards



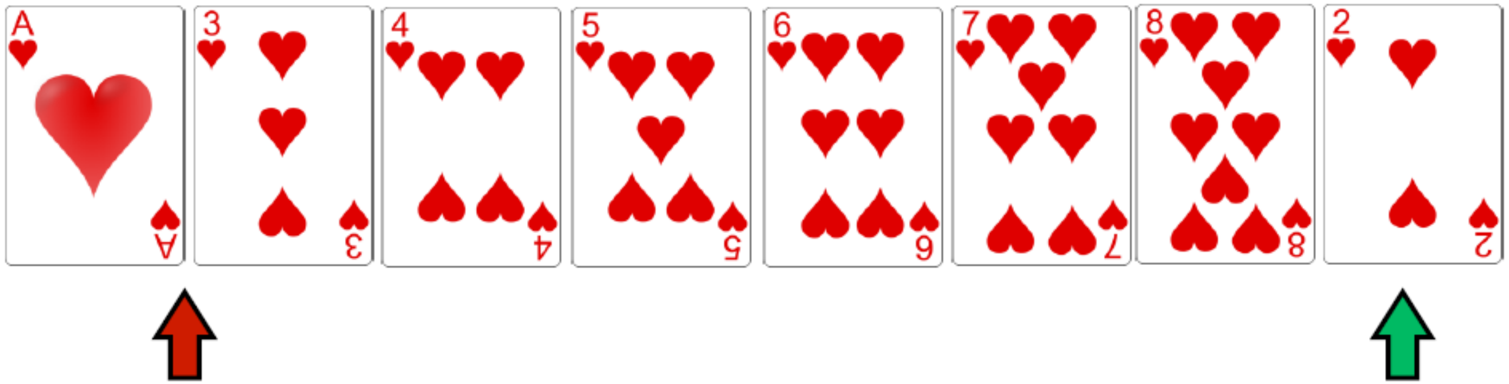
Insertion sort

- The seventh card (#8) is larger than the first six cards
- So, we don’t need to move it



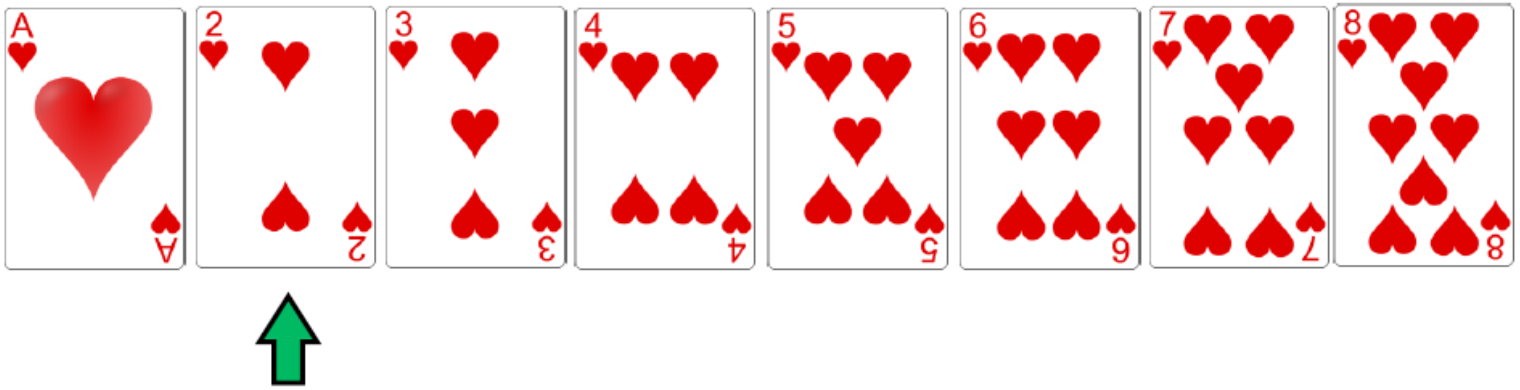
Insertion sort

- The eighth card (#2) should be inserted in between the first (#1) and second (#3) cards



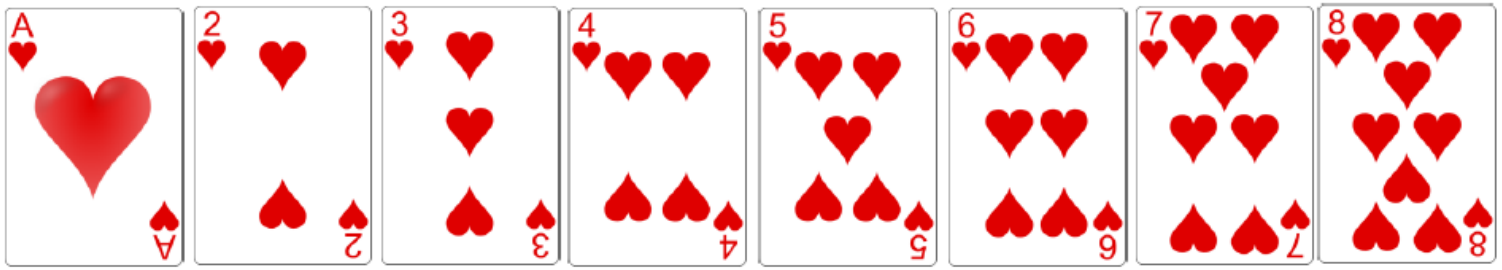
Insertion sort

- The eighth card (#2) should be inserted in between the first (#1) and second (#3) cards



Insertion sort

- The eighth card (#2) should be inserted in between the first (#1) and second (#3) cards



Finished!

Sorting algorithms

We saw there can be different ways to solve the same computational problem

- ➔ can derive many different algorithms for sorting

Algorithm:

- A set of concrete steps
- Steps solve a problem or accomplish some task
- Solve in a finite amount of time

The **Selection Sort** and **Insertion Sort** algorithms are only two ways of sorting a list of values

New problem: You want to sort a list of student records by their GPAs.

- Would both of these algorithms work?
- Yes! A sign of a good algorithm is that it is **general** ➔ can solve a variety of similar problems

Limits of Computation

What a computer can do:	What a computer cannot do:
Send email to a person if email address is known.	Find email of a person we met at a coffee shop.
Calculate different investment options based on historical data.	Choose a perfect investment or predict the future of companies.
Find information about colleges offering computer science courses.	Make a perfect decision on the best school to attend.
Solve well defined problems.	Solve ambiguous problems.

Intractable Problems

If a computer tries to analyze every possible sequence of moves in response to this opening in a game of chess, it will have to consider over 10^{43} games.

If a computer could solve one trillion combinations per second, it will compute the perfect game of chess if we are patient enough to wait 10^{21} years.

To sum up…

Computer science is the discipline of how to solve problems using computers

We strive for efficient, general solutions that will work on a wide variety of problems

Although computer science has existed as a field for about 70 years, its roots in mathematics and computation go back thousands of years!

CS is a field that relies partly on old mathematical ideas but experiences advances in development of new techniques at an extraordinary pace

This semester you will be exposed to many of the modern topics in CS and some of the older mathematical content that is still very relevant today