Introduction to Computational and Algorithmic Thinking

LECTURE 1 – COMPUTATIONAL THINKING

Announcements

This lecture: Computational Thinking

Reading: Read Chapter 1 of Conery

Acknowledgement: These slides are extended version of slides prepared by Prof. Arthur Lee, Prof. Tony Mione for earlier CSE101 classes.

What is Computer Science

- **Computer science** is all about using computers and computing technology to solve challenging, real-world problems in science, medicine, business and society (for us for now)
- 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
  - a fast track to becoming a nerd

Are you a good fit for CS?

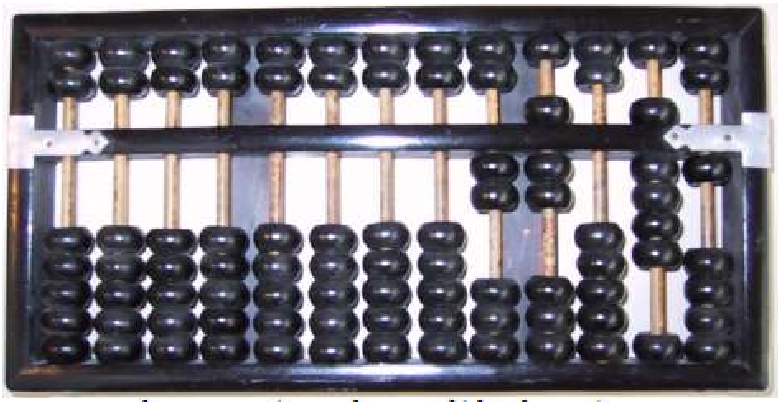
- You are a good fit for computer science if:
  - You are naturally curious and inquisitive.
  - You feel compelled to solve problems and puzzles.
  - You have a creative spirit and like making things.
  - You think in a logical, step-by-step manner.
  - You approach issues from unconventional angles.
  - You are willing to evolve and learn new things every day.
  - You are self-driven and have enough grit to endure long periods of frustration.
  - You know how to search the web for answers.
- List courtesy http://www.makeuseof.com/tag/what-is-computerscience

A modern computing problem

- Electronic health records are becoming increasingly important as time goes on
- Consider issues (technical + others) that arise providing a hw/sw 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 quick history of computing

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



- https://www.youtube.com/watch?v=GF6nCmcQ5es
- 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

1. Storage:
   - an abacus only 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)

A quick history of computing

2. Data Representation
   - the abacus represents numbers using beads on spindles.
   - Modern computers employ a variety of techniques for representing data on a variety of storage media
     - Magnetic
     - Optical
     - Electrical
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

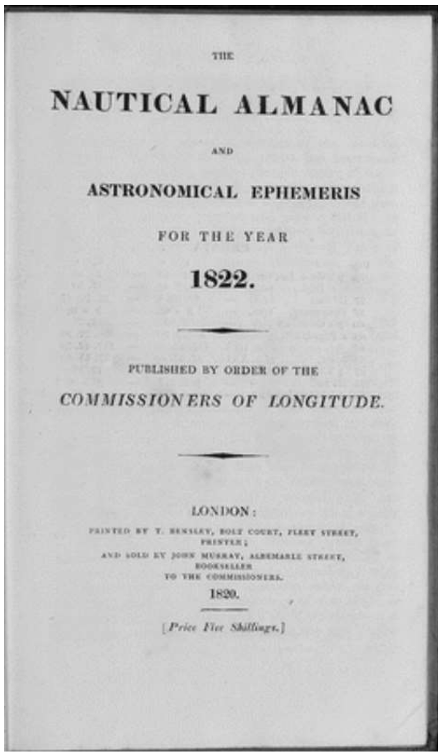
- In the 17th century people began tinkering with physical devices that could do computations and calculations
- Blaise Pascal
  - the French mathematician and philosopher
  - one of a few to design and build a physical calculator
- Calculator could only do addition and subtraction
  - Input is given 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
- This is not all that different from quitting a program that's running on your computer and starting another one!

- https://www.youtube.com/watch?v=MQzpLLhN0fY

Rise of Analytical Engine


THE

NAUTICAL ALMANAC

AND

ASTRONOMICAL EPHEMERIS

FOR THE YEAR

1822.

PUBLISHED BY ORDER OF THE

COMMISSIONERS OF LONGITUDE.

LONDON:

PRINTED BY T. BENSLEY, BOLT COURT, FLEET STREET, PRINTER;

AND SOLD BY JOHN MURRAY, ALBEMARLE STREET, BOOKSELLER TO THE COMMISSIONERS.

1820.

[*Price Five Shillings.*]

Summer of 1821 – Mathematician Charles Babbage and astronomer John Herschel were working on creating a book of mathematical tables.

Almanac contains tables denoting positions of the Moons, planets and stars – which are used by navigators to determine location at the sea.

Manual work caused a number of errors.

Babbage showed his frustration with the large number of errors by exclaiming, “I wish to God these calculations had been executed by steam!”

What made Babbage think steam engines could help him solve mathematical problems?

UT (d h)	ARIES GHA	VENUS −3.9 GHA	VENUS Dec	MARS +1.7 GHA	MARS Dec	JUPITER −2.0 GHA	JUPITER Dec	SATURN +0.1 GHA	SATURN Dec
10 00 (FRIDAY)	227 37.2	151 33.2	N23 58.3	151 02.2	N23 44.9	124 05.5	N23 04.8	154 11.5	N21 13.0
01	242 39.7	166 32.3	58.7	166 02.9	45.1	139 07.5	04.7	169 13.6	13.1
02	257 42.1	181 31.5	59.1	181 03.5	45.3	154 09.5	04.7	184 15.8	13.1
03	272 44.6	196 30.7	. . 59.5	196 04.2	. . 45.4	169 11.5	. . 04.7	199 17.9	. . 13.1
04	287 47.1	211 29.9	23 59.9	211 04.8	45.6	184 13.5	04.6	214 20.1	13.2
05	302 49.5	226 29.1	24 00.3	226 05.5	45.8	199 15.5	04.6	229 22.2	13.2
06	317 52.0	241 28.3	N24 00.6	241 06.1	N23 45.9	214 17.5	N23 04.5	244 24.3	N21 13.3
07	332 54.4	256 27.5	01.0	256 06.8	46.1	229 19.4	04.5	259 26.5	13.3
08	347 56.9	271 26.6	01.4	271 07.4	46.3	244 21.4	04.5	274 28.6	13.3
09	2 59.4	286 25.8	. . 01.8	286 08.1	. . 46.4	259 23.4	. . 04.4	289 30.8	. . 13.4
10	18 01.8	301 25.0	02.2	301 08.7	46.6	274 25.4	04.4	304 32.9	13.4
11	33 04.3	316 24.2	02.5	316 09.4	46.8	289 27.4	04.4	319 35.0	13.5
12	48 06.8	331 23.4	N24 02.9	331 10.0	N23 46.9	304 29.4	N23 04.3	334 37.2	N21 13.5
13	63 09.2	346 22.6	03.3	346 10.7	47.1	319 31.4	04.3	349 39.3	13.5
14	78 11.7	1 21.7	03.7	1 11.3	47.3	334 33.4	04.2	4 41.4	13.6
15	93 14.2	16 20.9	. . 04.0	16 12.0	. . 47.4	349 35.4	. . 04.2	19 43.6	. . 13.6
16	108 16.6	31 20.1	04.4	31 12.6	47.6	4 37.4	04.2	34 45.7	13.7
17	123 19.1	46 19.3	04.8	46 13.3	47.7	19 39.4	04.1	49 47.9	13.7
18	138 21.5	61 18.5	N24 05.1	61 13.9	N23 47.9	34 41.3	N23 04.1	64 50.0	N21 13.7
19	153 24.0	76 17.6	05.5	76 14.6	48.1	49 43.3	04.1	79 52.1	13.8
20	168 26.5	91 16.8	05.9	91 15.2	48.2	64 45.3	04.0	94 54.3	13.8
21	183 28.9	106 16.0	. . 06.2	106 15.9	. . 48.4	79 47.3	. . 04.0	109 56.4	. . 13.9
22	198 31.4	121 15.2	06.6	121 16.5	48.6	94 49.3	03.9	124 58.6	13.9
23	213 33.9	136 14.4	07.0	136 17.1	48.7	109 51.3	03.9	140 00.7	13.9
11 00 (SATURDAY)	228 36.3	151 13.5	N24 07.3	151 17.8	N23 48.9	124 53.3	N23 03.9	155 02.8	N21 14.0
01	243 38.8	166 12.7	07.7	166 18.4	49.0	139 55.3	03.8	170 05.0	14.0
02	258 41.3	181 11.9	08.0	181 19.1	49.2	154 57.3	03.8	185 07.1	14.1
03	273 43.7	196 11.1	. . 08.4	196 19.7	. . 49.4	169 59.3	. . 03.7	200 09.3	. . 14.1
04	288 46.2	211 10.3	08.8	211 20.4	49.5	185 01.2	03.7	215 11.4	14.1
05	303 48.7	226 09.4	09.1	226 21.0	49.7	200 03.2	03.7	230 13.5	14.2
06	318 51.1	241 08.6	N24 09.5	241 21.7	N23 49.8	215 05.2	N23 03.6	245 15.7	N21 14.2
07	333 53.6	256 07.8	09.8	256 22.3	50.0	230 07.2	03.6	260 17.8	14.3
08	348 56.0	271 07.0	10.2	271 23.0	50.1	245 09.2	03.6	275 19.9	14.3
09	3 58.5	286 06.2	. . 10.5	286 23.6	. . 50.3	260 11.2	. . 03.5	290 22.1	. . 14.3
10	19 01.0	301 05.3	10.9	301 24.3	50.5	275 13.2	03.5	305 24.2	14.4
11	34 03.4	316 04.5	11.2	316 24.9	50.6	290 15.2	03.4	320 26.4	14.4
12	49 05.9	331 03.7	N24 11.6	331 25.6	N23 50.8	305 17.1	N23 03.4	335 28.5	N21 14.5
13	64 08.4	346 02.9	11.9	346 26.2	50.9	320 19.1	03.4	350 30.6	14.5
14	79 10.8	1 02.0	12.3	1 26.9	51.1	335 21.1	03.3	5 32.8	14.5
15	94 13.3	16 01.2	. . 12.6	16 27.5	. . 51.2	350 23.1	. . 03.3	20 34.9	. . 14.6
16	109 15.8	31 00.4	12.9	31 28.2	51.4	5 25.1	03.2	35 37.1	14.6
17	124 18.2	45 59.6	13.3	46 28.8	51.5	20 27.1	03.2	50 39.2	14.7
18	139 20.7	60 58.8	N24 13.6	61 29.5	N23 51.7	35 29.1	N23 03.2	65 41.3	N21 14.7
19	154 23.1	75 57.9	14.0	76 30.1	51.9	50 31.1	03.1	80 43.5	14.7
20	169 25.6	90 57.1	14.3	91 30.8	52.0	65 33.0	03.1	95 45.6	14.8
21	184 28.1	105 56.3	. . 14.6	106 31.4	. . 52.2	80 35.0	. . 03.0	110 47.7	. . 14.8
22	199 30.5	120 55.5	15.0	121 32.1	52.3	95 37.0	03.0	125 49.9	14.9
23	214 33.0	135 54.6	15.3	136 32.7	52.5	110 39.0	03.0	140 52.0	14.9
12 00 (SUNDAY)	229 35.5	150 53.8	N24 15.6	151 33.4	N23 52.6	125 41.0	N23 02.9	155 54.2	N21 14.9
01	244 37.9	165 53.0	16.0	166 34.0	52.8	140 43.0	02.9	170 56.3	15.0
02	259 40.4	180 52.2	16.3	181 34.7	52.9	155 45.0	02.9	185 58.4	15.0
03	274 42.9	195 51.3	. . 16.6	196 35.3	. . 53.1	170 46.9	. . 02.8	201 00.6	. . 15.0
04	289 45.3	210 50.5	17.0	211 35.9	53.2	185 48.9	02.8	216 02.7	15.1
05	304 47.8	225 49.7	17.3	226 36.6	53.4	200 50.9	02.7	231 04.8	15.1
06	319 50.3	240 48.9	N24 17.6	241 37.2	N23 53.5	215 52.9	N23 02.7	246 07.0	N21 15.2
07	334 52.7	255 48.0	17.9	256 37.9	53.7	230 54.9	02.7	261 09.1	15.2
08	349 55.2	270 47.2	18.3	271 38.5	53.8	245 56.9	02.6	276 11.2	15.2
09	4 57.6	285 46.4	. . 18.6	286 39.2	. . 54.0	260 58.8	. . 02.6	291 13.4	. . 15.3
10	20 00.1	300 45.5	18.9	301 39.8	54.1	276 00.8	02.5	306 15.5	15.3
11	35 02.6	315 44.7	19.2	316 40.5	54.3	291 02.8	02.5	321 17.7	15.4
12	50 05.0	330 43.9	N24 19.5	331 41.1	N23 54.4	306 04.8	N23 02.5	336 19.8	N21 15.4
13	65 07.5	345 43.1	19.9	346 41.8	54.6	321 06.8	02.4	351 21.9	15.4
14	80 10.0	0 42.2	20.2	1 42.4	54.7	336 08.8	02.4	6 24.1	15.5
15	95 12.4	15 41.4	. . 20.5	16 43.1	. . 54.8	351 10.7	. . 02.3	21 26.2	. . 15.5
16	110 14.9	30 40.6	20.8	31 43.7	55.0	6 12.7	02.3	36 28.3	15.6
17	125 17.4	45 39.8	21.1	46 44.4	55.1	21 14.7	02.3	51 30.5	15.6
18	140 19.8	60 38.9	N24 21.4	61 45.0	N23 55.3	36 16.7	N23 02.2	66 32.6	N21 15.6
19	155 22.3	75 38.1	21.7	76 45.7	55.4	51 18.7	02.2	81 34.7	15.7
20	170 24.8	90 37.3	22.0	91 46.3	55.6	66 20.7	02.1	96 36.9	15.7
21	185 27.2	105 36.4	. . 22.3	106 47.0	. . 55.7	81 22.6	. . 02.1	111 39.0	. . 15.8
22	200 29.7	120 35.6	22.6	121 47.6	55.9	96 24.6	02.1	126 41.2	15.8
23	215 32.1	135 34.8	23.0	136 48.3	56.0	111 26.6	02.0	141 43.3	15.8
Mer. Pass.	h m 8 44.1	v −0.8	d 0.3	v 0.6	d 0.2	v 2.0	d 0.0	v 2.1	d 0.0

STARS Name	SHA	Dec
Acamar	315 25.3	S40 17.8
Achernar	335 33.7	S57 13.5
Acrux	173 18.8	S63 06.9
Adhara	255 19.7	S28 58.7
Aldebaran	290 59.8	N16 30.8
Alioth	166 27.8	N55 57.1
Alkaid	153 05.2	N49 18.2
Al Na'ir	27 54.7	S46 56.9
Alnilam	275 55.6	S 1 12.1
Alphard	218 04.7	S 8 40.2
Alphecca	126 18.0	N26 42.4
Alpheratz	357 52.9	N29 05.9
Altair	62 16.6	N 8 52.3
Ankaa	353 24.6	S42 17.6
Antares	112 36.7	S26 26.2
Arcturus	146 03.4	N19 10.3
Atria	107 46.0	S69 01.8
Avior	234 21.9	S59 31.3
Bellatrix	278 41.7	N 6 21.0
Betelgeuse	271 11.1	N 7 24.4
Canopus	264 00.4	S52 42.0
Capella	280 47.9	N46 00.1
Deneb	49 37.4	N45 16.9
Denebola	182 42.4	N14 33.6
Diphda	349 04.9	S17 58.6
Dubhe	194 01.9	N61 44.6
Elnath	278 24.1	N28 36.6
Eltanin	90 49.8	N51 29.1
Enif	33 55.8	N 9 52.9
Fomalhaut	15 33.7	S29 36.6
Gacrux	172 10.5	S57 07.7
Gienah	176 01.1	S17 33.3
Hadar	148 59.9	S60 23.1
Hamal	328 11.1	N23 28.2
Kaus Aust.	83 55.1	S34 23.0
Kochab	137 18.2	N74 08.9
Markab	13 47.2	N15 12.8
Menkar	314 24.6	N 4 05.8
Menkent	148 17.6	S36 22.9
Miaplacidus	221 41.8	S69 43.8
Mirfak	308 53.5	N49 52.1
Nunki	76 08.9	S26 17.6
Peacock	53 32.7	S56 43.5
Pollux	243 38.6	N28 01.4
Procyon	245 09.1	N 5 13.1
Rasalhague	96 14.3	N12 33.4
Regulus	207 52.8	N11 57.4
Rigel	281 20.8	S 8 12.0
Rigil Kent.	140 03.3	S60 50.7
Sabik	102 22.3	S15 43.7
Schedar	349 51.2	N56 32.7
Shaula	96 33.5	S37 06.3
Sirius	258 41.7	S16 43.3
Spica	158 40.2	S11 10.4
Suhail	222 59.0	S43 26.7
Vega	80 44.6	N38 46.9
Zuben'ubi	137 14.8	S16 03.1

	SHA	Mer. Pass.
	° ′	h m
Venus	282 37.2	13 56
Mars	282 41.5	13 54
Jupiter	256 17.0	15 38
Saturn	286 26.5	13 38

UT (d h)	SUN GHA	SUN Dec	MOON GHA	v	MOON Dec	d	HP
10 00 (FRIDAY)	180 54.3	N17 31.0	205 56.8	16.1	N 4 25.8	12.5	54.4
01	195 54.3	31.6	220 31.9	16.1	4 38.3	12.4	54.4
02	210 54.3	32.3	235 07.0	16.0	4 50.7	12.5	54.4
03	225 54.3	. . 32.9	249 42.0	16.0	5 03.2	12.5	54.4
04	240 54.4	33.6	264 17.0	16.0	5 15.7	12.4	54.4
05	255 54.4	34.3	278 52.0	16.0	5 28.1	12.5	54.4
06	270 54.4	N17 34.9	293 27.0	15.9	N 5 40.6	12.4	54.5
07	285 54.4	35.6	308 01.9	15.9	5 53.0	12.4	54.5
08	300 54.5	36.2	322 36.8	15.8	6 05.4	12.4	54.5
09	315 54.5	. . 36.9	337 11.6	15.8	6 17.8	12.3	54.5
10	330 54.5	37.5	351 46.4	15.8	6 30.1	12.4	54.5
11	345 54.5	38.2	6 21.2	15.7	6 42.5	12.3	54.5
12	0 54.5	N17 38.8	20 55.9	15.7	N 6 54.8	12.3	54.5
13	15 54.6	39.5	35 30.6	15.7	7 07.1	12.3	54.5
14	30 54.6	40.2	50 05.3	15.6	7 19.4	12.3	54.6
15	45 54.6	. . 40.8	64 39.9	15.5	7 31.7	12.2	54.6
16	60 54.6	41.5	79 14.4	15.6	7 43.9	12.3	54.6
17	75 54.7	42.1	93 49.0	15.4	7 56.2	12.2	54.6
18	90 54.7	N17 42.8	108 23.4	15.5	N 8 08.4	12.1	54.6
19	105 54.7	43.4	122 57.9	15.4	8 20.5	12.2	54.6
20	120 54.7	44.1	137 32.3	15.3	8 32.7	12.1	54.6
21	135 54.7	. . 44.7	152 06.6	15.3	8 44.8	12.1	54.7
22	150 54.8	45.4	166 40.9	15.3	8 56.9	12.1	54.7
23	165 54.8	46.0	181 15.2	15.2	9 09.0	12.0	54.7
11 00 (SATURDAY)	180 54.8	N17 46.6	195 49.4	15.2	N 9 21.0	12.0	54.7
01	195 54.8	47.3	210 23.6	15.1	9 33.0	12.0	54.7
02	210 54.8	47.9	224 57.7	15.0	9 45.0	11.9	54.7
03	225 54.9	. . 48.6	239 31.7	15.0	9 56.9	11.9	54.7
04	240 54.9	49.2	254 05.7	15.0	10 08.8	11.9	54.8
05	255 54.9	49.9	268 39.7	14.9	10 20.7	11.8	54.8
06	270 54.9	N17 50.5	283 13.6	14.9	N10 32.5	11.8	54.8
07	285 54.9	51.2	297 47.5	14.8	10 44.3	11.8	54.8
08	300 54.9	51.8	312 21.3	14.7	10 56.1	11.7	54.8
09	315 55.0	. . 52.5	326 55.0	14.7	11 07.8	11.7	54.8
10	330 55.0	53.1	341 28.7	14.6	11 19.5	11.7	54.9
11	345 55.0	53.7	356 02.3	14.6	11 31.2	11.6	54.9
12	0 55.0	N17 54.4	10 35.9	14.5	N11 42.8	11.6	54.9
13	15 55.0	55.0	25 09.4	14.4	11 54.4	11.5	54.9
14	30 55.0	55.7	39 42.8	14.4	12 05.9	11.5	54.9
15	45 55.0	. . 56.3	54 16.2	14.4	12 17.4	11.5	54.9
16	60 55.1	56.9	68 49.6	14.2	12 28.9	11.4	54.9
17	75 55.1	57.6	83 22.8	14.3	12 40.3	11.3	55.0
18	90 55.1	N17 58.2	97 56.1	14.1	N12 51.6	11.3	55.0
19	105 55.1	58.8	112 29.2	14.1	13 02.9	11.3	55.0
20	120 55.1	17 59.5	127 02.3	14.0	13 14.2	11.2	55.0
21	135 55.1	18 00.1	141 35.3	14.0	13 25.4	11.1	55.0
22	150 55.1	00.8	156 08.3	13.9	13 36.5	11.2	55.0
23	165 55.2	01.4	170 41.2	13.8	13 47.7	11.0	55.1
12 00 (SUNDAY)	180 55.2	N18 02.0	185 14.0	13.7	N13 58.7	11.0	55.1
01	195 55.2	02.7	199 46.7	13.7	14 09.7	11.0	55.1
02	210 55.2	03.3	214 19.4	13.7	14 20.7	10.9	55.1
03	225 55.2	. . 03.9	228 52.1	13.5	14 31.6	10.8	55.1
04	240 55.2	04.6	243 24.6	13.5	14 42.4	10.8	55.2
05	255 55.2	05.2	257 57.1	13.4	14 53.2	10.7	55.2
06	270 55.2	N18 05.8	272 29.5	13.4	N15 03.9	10.7	55.2
07	285 55.3	06.5	287 01.9	13.3	15 14.6	10.6	55.2
08	300 55.3	07.1	301 34.2	13.2	15 25.2	10.5	55.2
09	315 55.3	. . 07.7	316 06.4	13.1	15 35.7	10.5	55.2
10	330 55.3	08.3	330 38.5	13.1	15 46.2	10.5	55.3
11	345 55.3	09.0	345 10.6	13.0	15 56.7	10.3	55.3
12	0 55.3	N18 09.6	359 42.6	12.9	N16 07.0	10.3	55.3
13	15 55.3	10.2	14 14.5	12.8	16 17.3	10.2	55.3
14	30 55.3	10.9	28 46.3	12.8	16 27.5	10.2	55.3
15	45 55.3	. . 11.5	43 18.1	12.7	16 37.7	10.1	55.3
16	60 55.4	12.1	57 49.8	12.6	16 47.8	10.0	55.4
17	75 55.4	12.7	72 21.4	12.6	16 57.8	9.9	55.4
18	90 55.4	N18 13.4	86 53.0	12.4	N17 07.7	9.9	55.4
19	105 55.4	14.0	101 24.4	12.4	17 17.6	9.8	55.4
20	120 55.4	14.6	115 55.8	12.4	17 27.4	9.8	55.4
21	135 55.4	. . 15.2	130 27.2	12.2	17 37.2	9.6	55.5
22	150 55.4	15.9	144 58.4	12.2	17 46.8	9.6	55.5
23	165 55.4	16.5	159 29.6	12.1	N17 56.4	9.5	55.5
	SD 15.9	d 0.6	SD 14.9		15.0		15.1

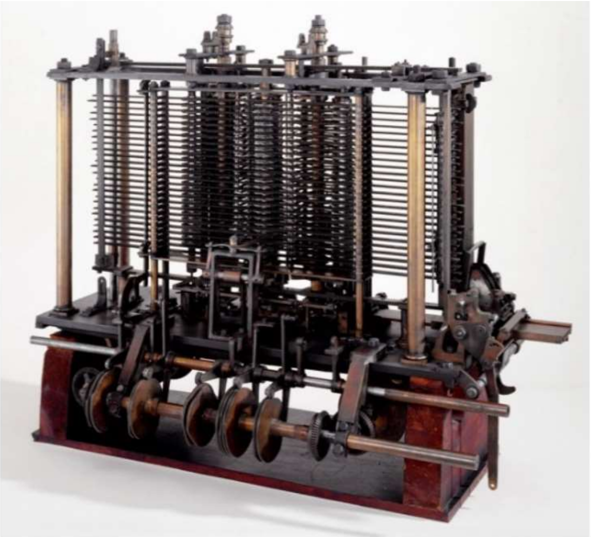
Lat.	Twilight Naut.	Twilight Civil	Sunrise	Moonrise 10	Moonrise 11	Moonrise 12	Moonrise 13
°	h m	h m	h m	h m	h m	h m	h m
N 72	▭	▭	▭	03 19	02 53	02 17	▭
N 70	////	////	01 25	03 26	03 08	02 45	02 06
68	////	////	02 10	03 31	03 20	03 06	02 47
66	////	00 28	02 39	03 36	03 30	03 23	03 15
64	////	01 35	03 01	03 40	03 38	03 37	03 37
62	////	02 09	03 18	03 44	03 46	03 49	03 55
60	00 25	02 33	03 33	03 47	03 52	03 59	04 09
N 58	01 25	02 52	03 45	03 50	03 58	04 08	04 22
56	01 56	03 07	03 55	03 52	04 03	04 16	04 32
54	02 19	03 20	04 05	03 54	04 07	04 23	04 42
52	02 36	03 32	04 13	03 56	04 11	04 29	04 51
50	02 51	03 42	04 20	03 58	04 15	04 35	04 58
45	03 20	04 02	04 36	04 02	04 23	04 47	05 15
N 40	03 42	04 19	04 49	04 06	04 30	04 57	05 28
35	03 59	04 32	05 00	04 09	04 36	05 06	05 40
30	04 12	04 44	05 09	04 11	04 41	05 14	05 50
20	04 35	05 02	05 26	04 16	04 50	05 27	06 07
N 10	04 52	05 18	05 40	04 20	04 58	05 39	06 23
0	05 06	05 31	05 53	04 24	05 06	05 50	06 37
S 10	05 18	05 44	06 06	04 28	05 14	06 01	06 52
20	05 30	05 56	06 20	04 32	05 22	06 13	07 07
30	05 41	06 10	06 35	04 37	05 31	06 27	07 25
35	05 47	06 17	06 44	04 40	05 37	06 35	07 36
40	05 53	06 25	06 54	04 43	05 43	06 45	07 48
45	05 59	06 35	07 06	04 46	05 50	06 56	08 02
S 50	06 06	06 46	07 21	04 51	05 59	07 09	08 20
52	06 09	06 50	07 28	04 53	06 03	07 15	08 28
54	06 13	06 56	07 35	04 55	06 08	07 22	08 38
56	06 16	07 02	07 43	04 58	06 13	07 30	08 48
58	06 20	07 08	07 53	05 01	06 18	07 38	09 00
S 60	06 24	07 15	08 04	05 04	06 24	07 48	09 15

Lat.	Sunset	Twilight Civil	Twilight Naut.	Moonset 10	Moonset 11	Moonset 12	Moonset 13
°	h m	h m	h m	h m	h m	h m	h m
N 72	▭	▭	▭	18 32	20 39	▭	▭
N 70	22 35	////	////	18 19	20 12	22 27	▭
68	21 47	////	////	18 09	19 52	21 47	24 15
66	21 17	////	////	18 00	19 37	21 20	23 14
64	20 54	22 23	////	17 53	19 24	20 59	22 40
62	20 37	21 47	////	17 47	19 13	20 43	22 15
60	20 22	21 22	////	17 42	19 04	20 29	21 55
N 58	20 09	21 03	22 32	17 37	18 56	20 17	21 39
56	19 59	20 47	22 00	17 33	18 49	20 07	21 25
54	19 49	20 34	21 36	17 30	18 43	19 58	21 14
52	19 41	20 22	21 18	17 26	18 37	19 50	21 03
50	19 33	20 12	21 03	17 23	18 32	19 43	20 54
45	19 17	19 51	20 34	17 17	18 21	19 27	20 35
N 40	19 04	19 35	20 12	17 12	18 12	19 15	20 19
35	18 53	19 21	19 55	17 07	18 05	19 04	20 05
30	18 44	19 10	19 41	17 03	17 58	18 55	19 54
20	18 27	18 51	19 18	16 56	17 46	18 39	19 34
N 10	18 13	18 35	19 01	16 50	17 36	18 25	19 17
0	18 00	18 22	18 47	16 44	17 27	18 12	19 01
S 10	17 47	18 09	18 34	16 38	17 17	18 00	18 45
20	17 33	17 56	18 23	16 32	17 07	17 46	18 28
30	17 17	17 42	18 11	16 25	16 56	17 30	18 09
35	17 08	17 35	18 05	16 21	16 50	17 21	17 57
40	16 58	17 27	17 59	16 17	16 42	17 11	17 45
45	16 46	17 17	17 53	16 11	16 33	16 59	17 29
S 50	16 31	17 07	17 46	16 05	16 23	16 44	17 11
52	16 25	17 02	17 43	16 02	16 18	16 38	17 02
54	16 17	16 56	17 39	15 59	16 13	16 30	16 52
56	16 09	16 50	17 36	15 56	16 07	16 22	16 41
58	15 59	16 44	17 32	15 52	16 01	16 12	16 28
S 60	15 48	16 36	17 28	15 48	15 54	16 02	16 14

Day	SUN Eqn. of Time 00^h	SUN Eqn. of Time 12^h	SUN Mer. Pass.	MOON Mer. Pass. Upper	MOON Mer. Pass. Lower	Age	Phase
d	m s	m s	h m	h m	h m	d	%
10	03 37	03 38	11 56	10 34	22 55	28	4
11	03 39	03 40	11 56	11 16	23 38	29	1
12	03 41	03 41	11 56	12 01	24 25	00	0

Programmable devices

- Charles Babbage designed the Analytical Engine, a mechanical, programmable computer in 19^{th} Century
  - It was never built in Babbage’s time due to a lack of manufacturing capabilities (ahead of his time!)
  - 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 punched cards

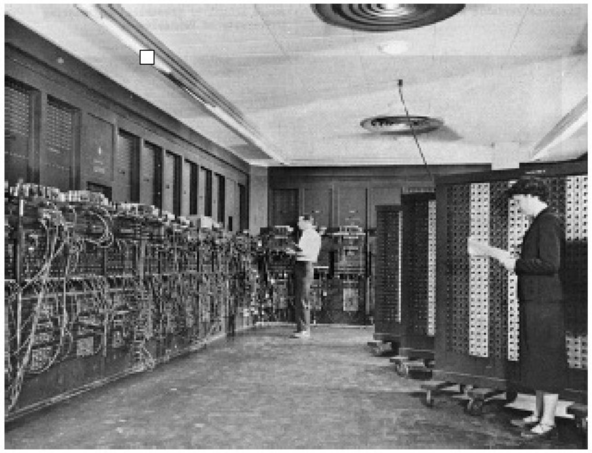


Programmable computers

- Many others
- Now, move forward to the 20th and 21st centuries
- A modern computer has three basic requirements:
  1. Must be electronic and not exclusively mechanical.
  2. Must be digital, not analog
     - Uses 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
- A modern computer has four major kinds of components:
  - Input device(s) – examples?
  - Output device(s) – examples?
  - 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 easily reprogram the computer to do new tasks

A typical processor has a thousand or more different machine instructions.

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

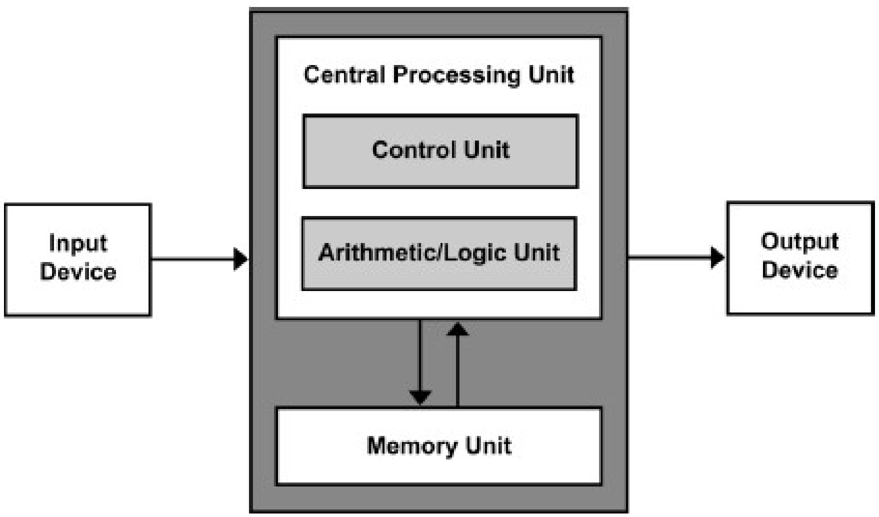
- Noyce and businessman Gordon Moore commercialized this technology by co-founding Intel Corporation in 1968
- Manufacturing technologies improved in the 1950s and 1960s:
  - engineers were able to pack many more transistors per unit area on silicon wafers
  - **Moore's law:** Moore observed that the number of components within an integrated circuit was doubling every 18 months.
    - The trend has continued pretty steadily since then.
    - 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

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, main memory
  - Hard drives and other storage units
- Type of software in use
  - Applications software, like office productivity programs, video games, web browsers
  - Systems software, like operating systems, database systems

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

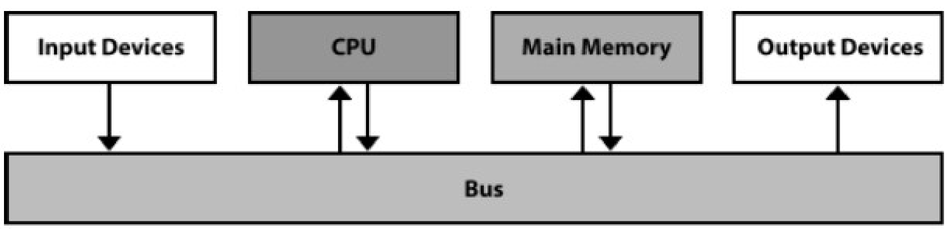
Modern computer architecture

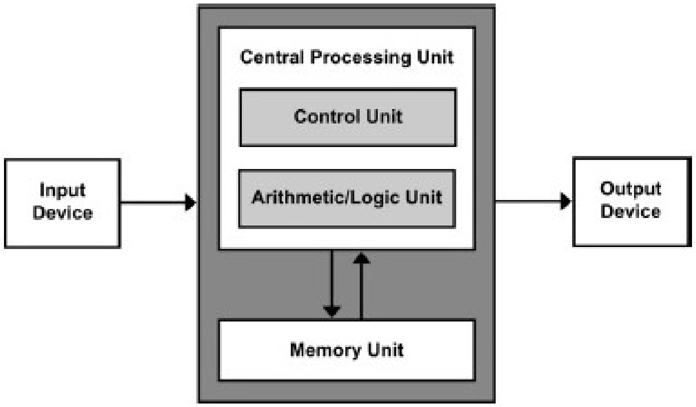
- In modern computers (PCs), the major components in a von Neumann machine reside physically in a circuit board called the **motherboard**
  - The CPU, memory, expansion cards and other components are plugged into slots so they can be replaced
  - Hard drives, CD drives, and other storage devices are connected to the motherboard through cables
- The central processing unit 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

Modern computer architecture

- The memory unit in this diagram refers to the main memory, not hard drives and other forms of **external storage**
- CPU, main memory, and I/O devices communicate over a shared set of wires known as the **system bus**



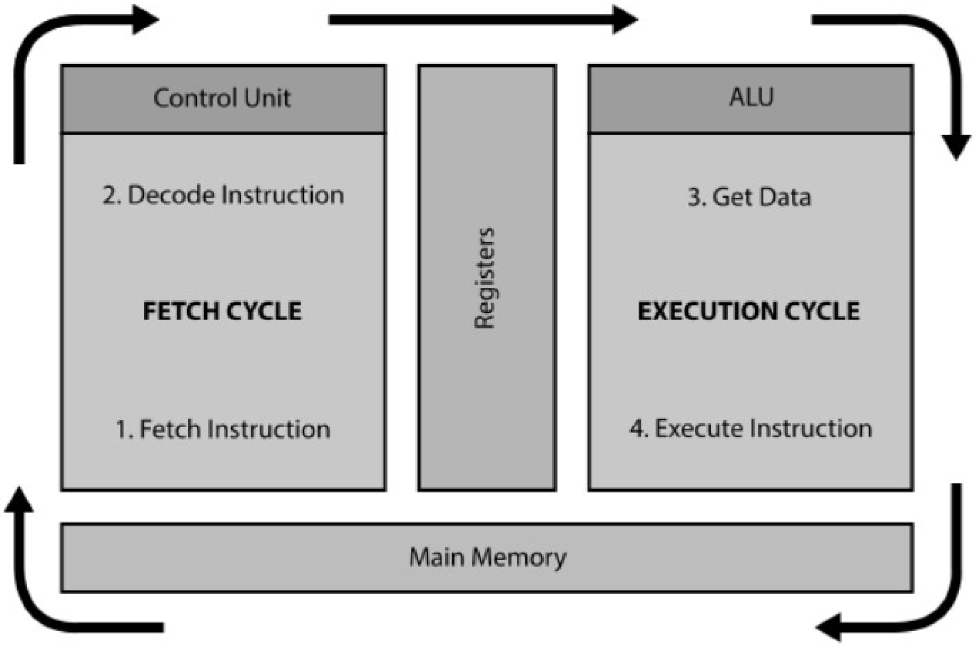


The fetch-decode-execute cycle

- The system bus carries 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, subtraction, whatever)
  - 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**
- Illustration

https://chortle.ccsu.edu/java5/Notes/chap04/ch04_4.html

The fetch-decode-execute cycle



What about the software?

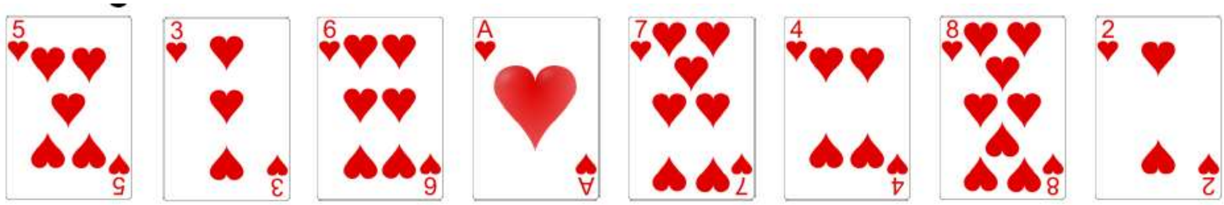
- Software consists of instructions for the CPU to execute
  - CPUs “understand” something called **machine language**, which consists of 0s and 1s
  - A single instruction for a modern computer might consist of some combination of 32 or 64 0s and 1s!
- Most programming now done using **high-level programming languages**, which consist of English and English-like words with some mathematical notation thrown in
- Will look into the basics of Python, a popular, easy-to-learn, high-level programming language

What is computational thinking?

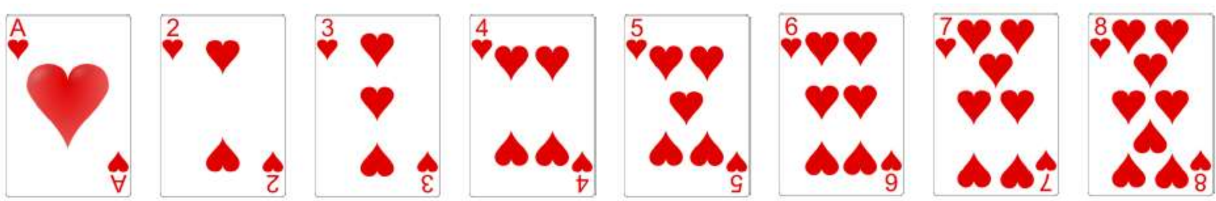
- **Computational Thinking:**
  - How computer scientists think – how they reason and work through problems
- Computer science encompasses many sub-disciplines that support the goal of solving problems:
  - Computer theory areas ➔ these are the heart and soul of computer science
    - algorithms
    - data structures
  - Computer systems areas
    - hardware design
    - operating systems
    - networks
  - Computer software and applications
    - software engineering
    - programming languages
    - Databases
    - Simulation
    - artificial intelligence
    - computer graphics
- A major goal of this course is to help you develop your computational thinking and problem solving skills

A classical problem: Sorting data

- Suppose we have a deck of cards we want to put in order
- **Sorting**: important problem – arises frequently in computer science
- Example: Use the Ace thru 8 of Hearts
- Given:

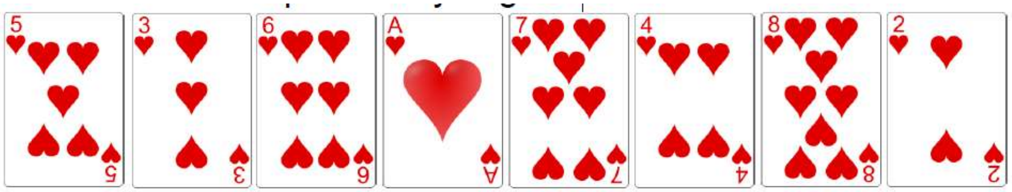


- Want the following:



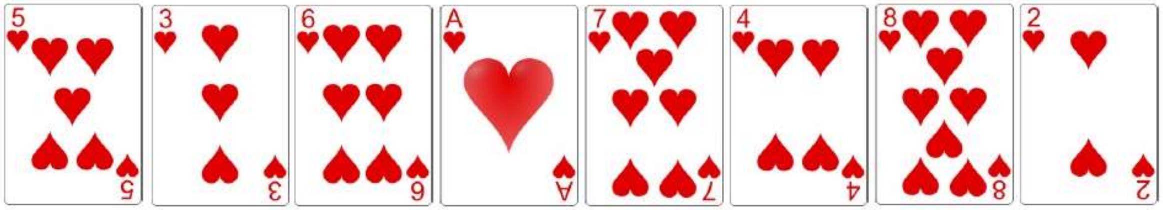
A classical problem: Sorting data

- You want to explain to a young child how to put the cards in order
  - What steps would you give?



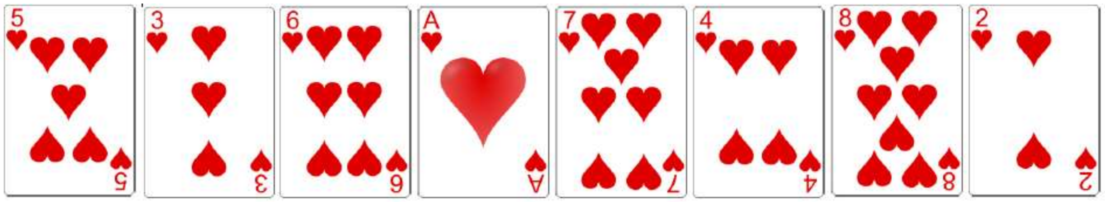
A classical problem: Sorting data

- One sorting technique is called **selection sort**
- It repeatedly searches for and swaps cards in the list



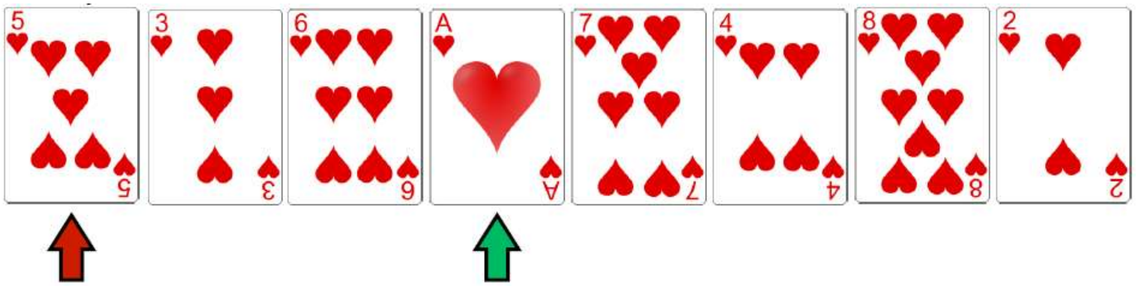
Selection sort

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



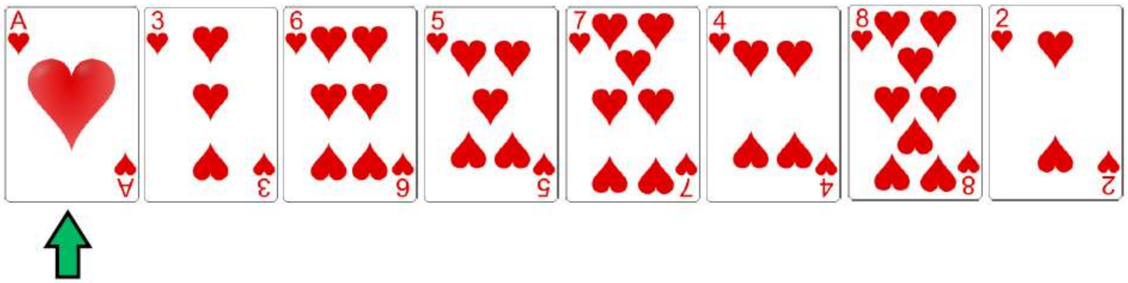
Selection sort

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



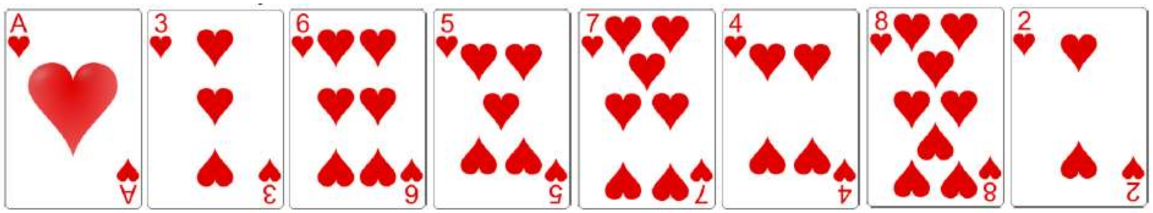
Selection sort

- Collection after the exchange



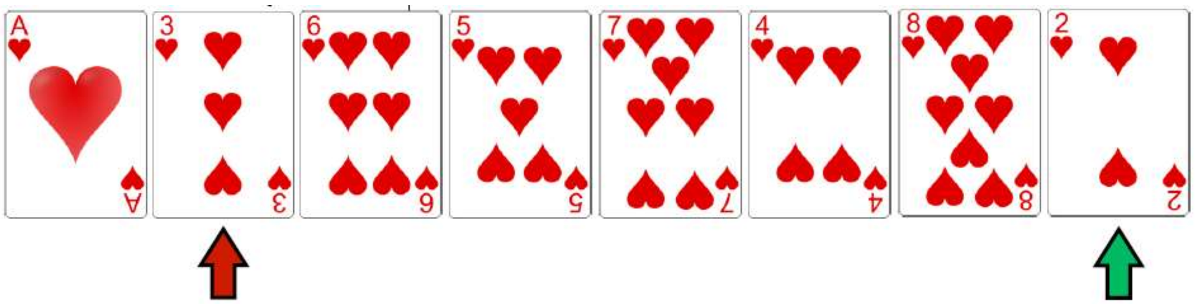
Selection sort

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



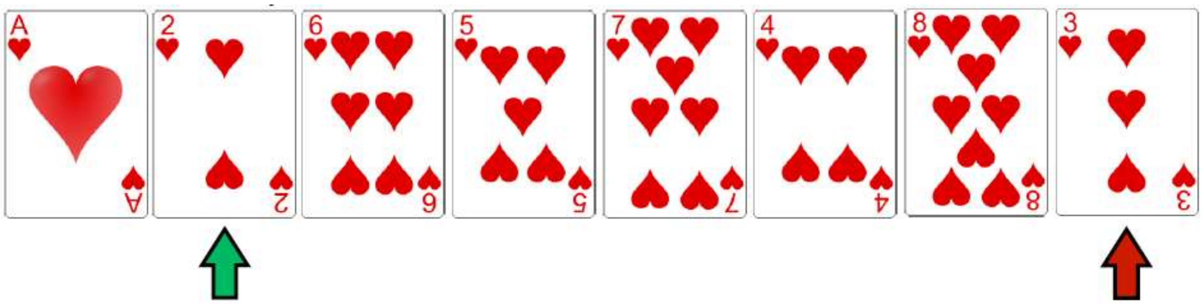
Selection sort

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



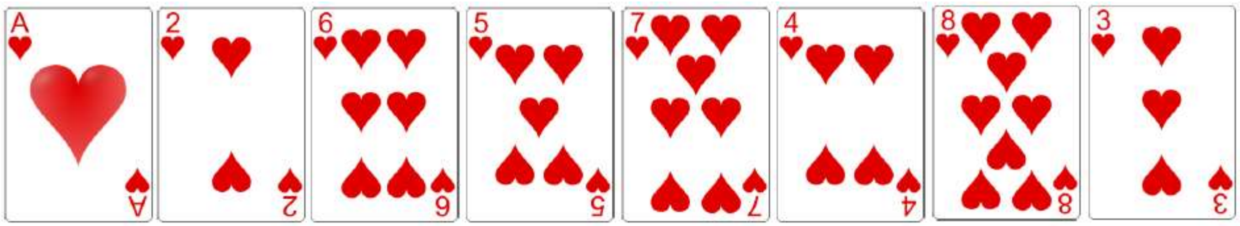
Selection sort

- Collection after the exchange



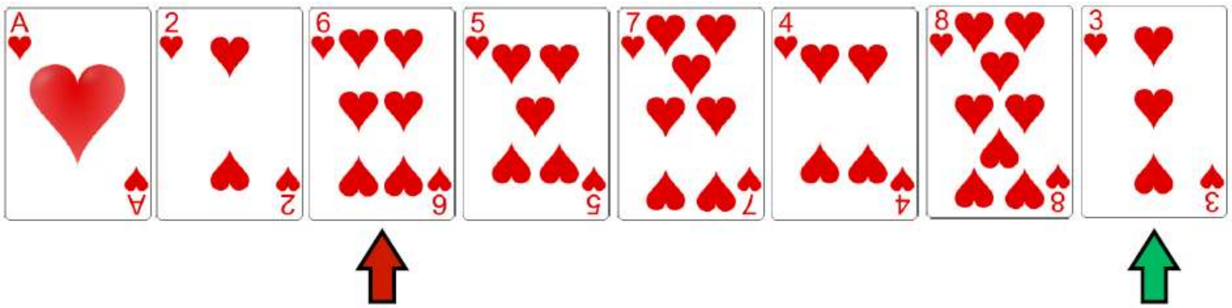
Selection sort

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



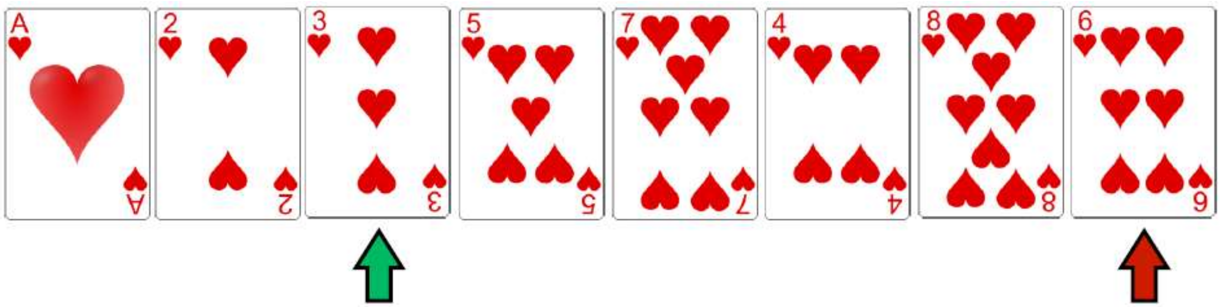
Selection sort

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



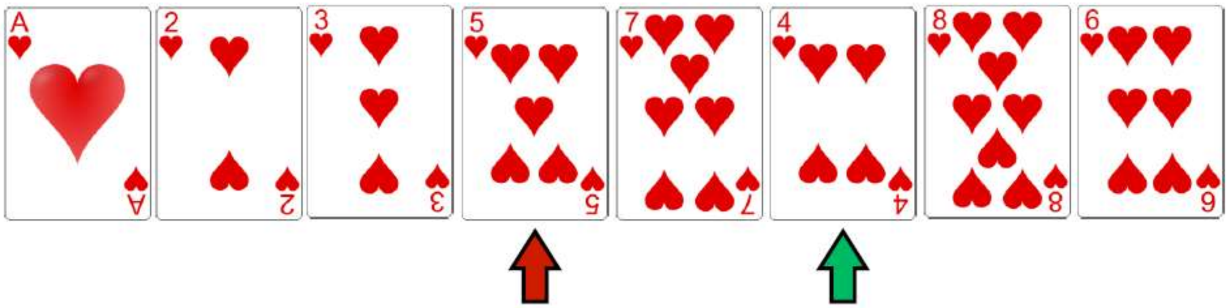
Selection sort

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



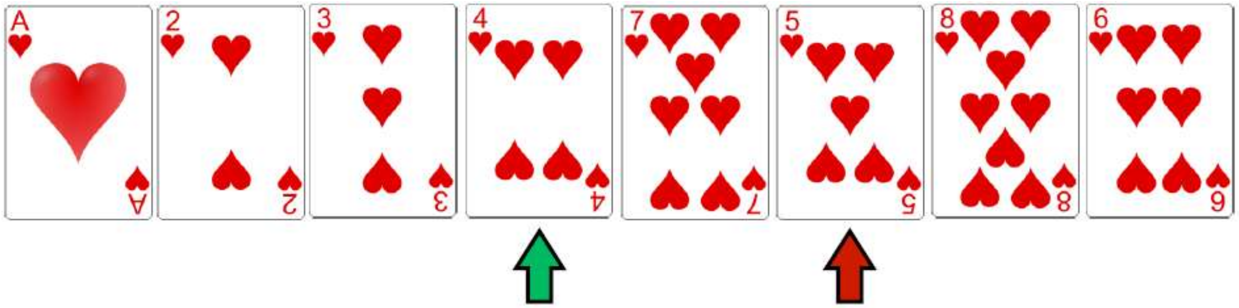
Selection sort

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



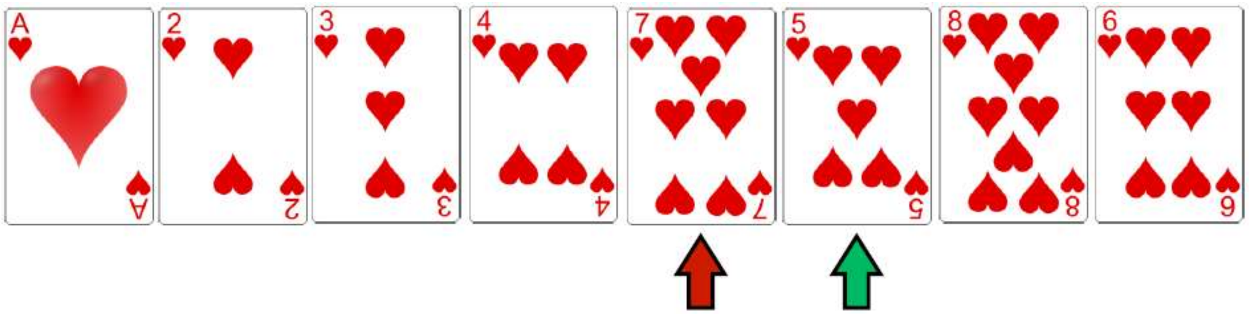
Selection sort

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



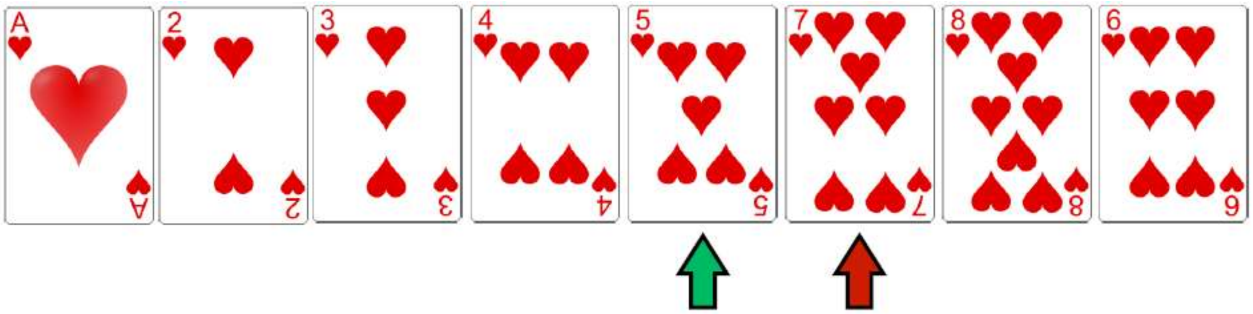
Selection sort

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



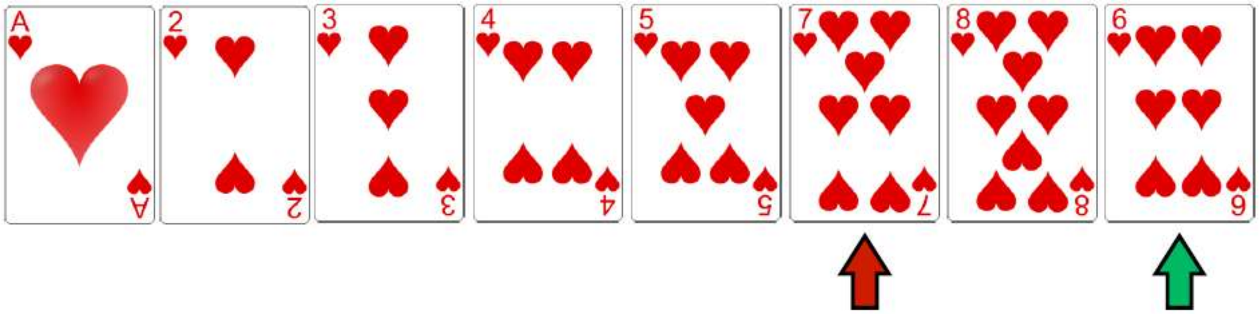
Selection sort

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



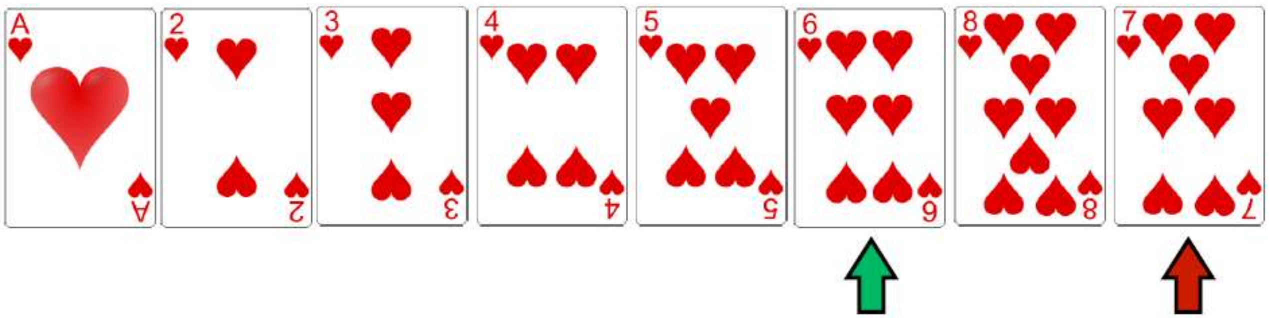
Selection sort

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



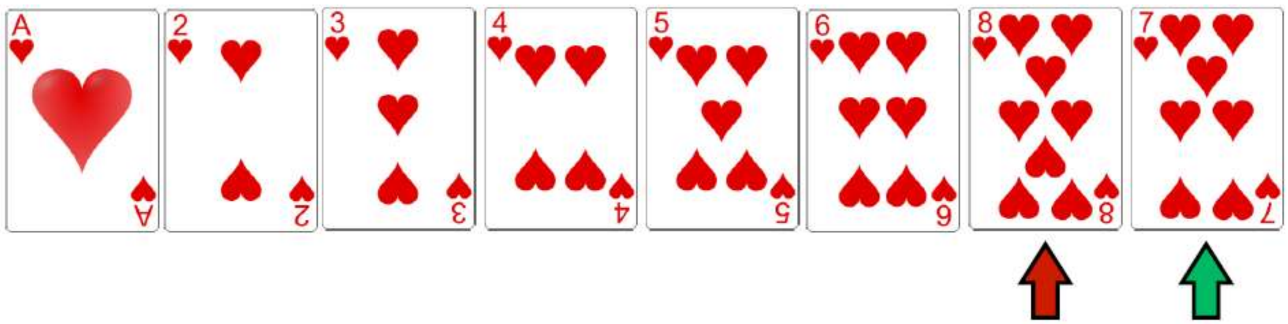
Selection sort

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



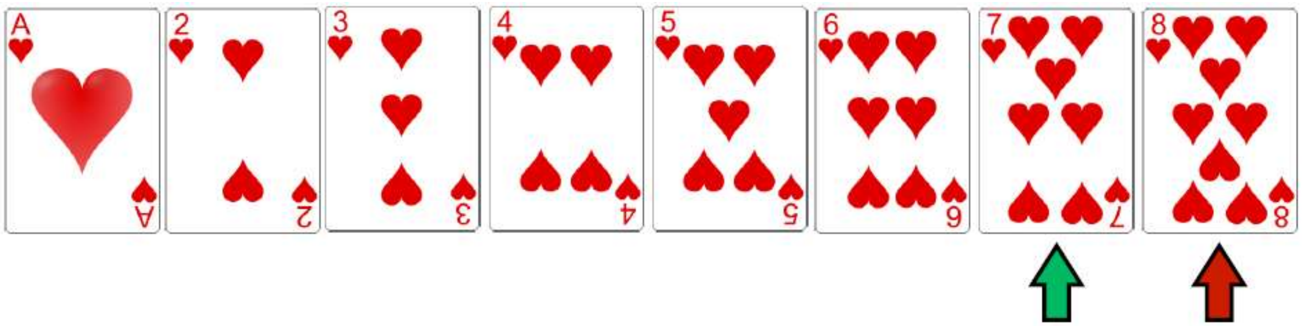
Selection sort

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



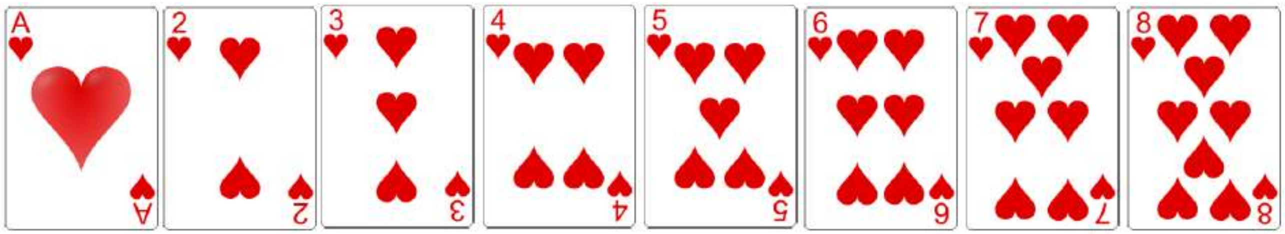
Selection sort

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



Selection sort

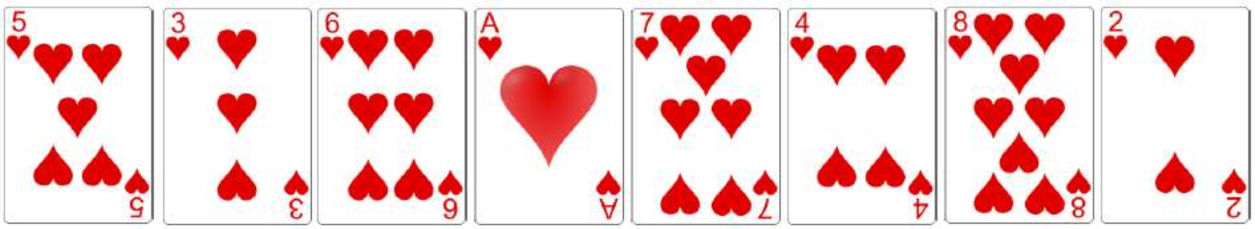
- Continue in this fashion, *selecting* the third-smallest, fourth-smallest, etc., until the list is sorted



Finished!

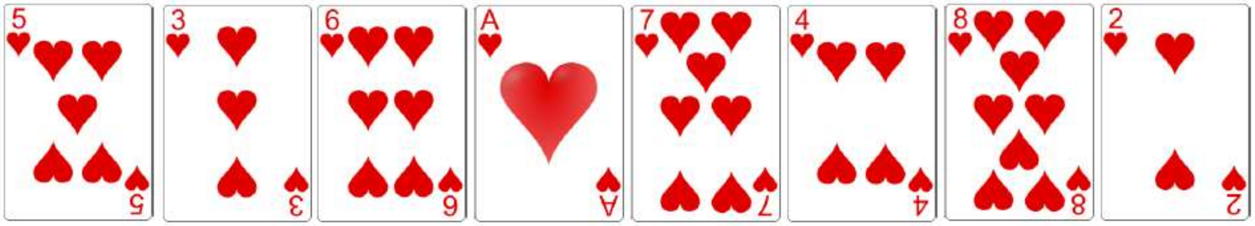
A classical problem: sorting data (2)

- Another sorting technique is **insertion sort**
- It repeatedly inserts the “next” card into its correct spot



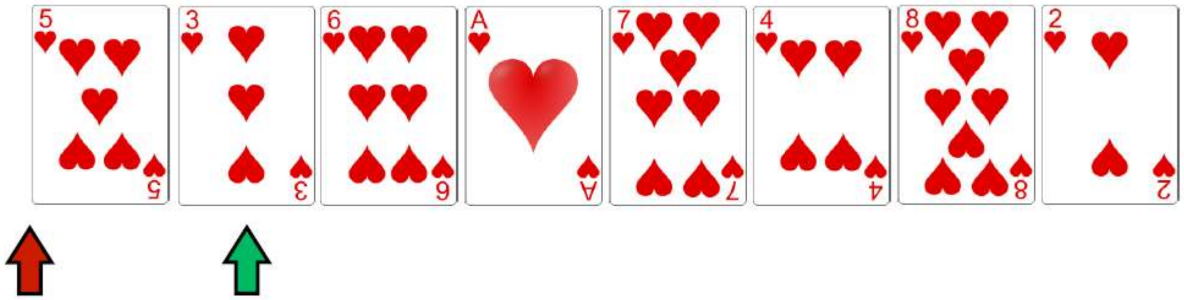
Insertion sort

- We begin by leaving the first card (#5) where it is



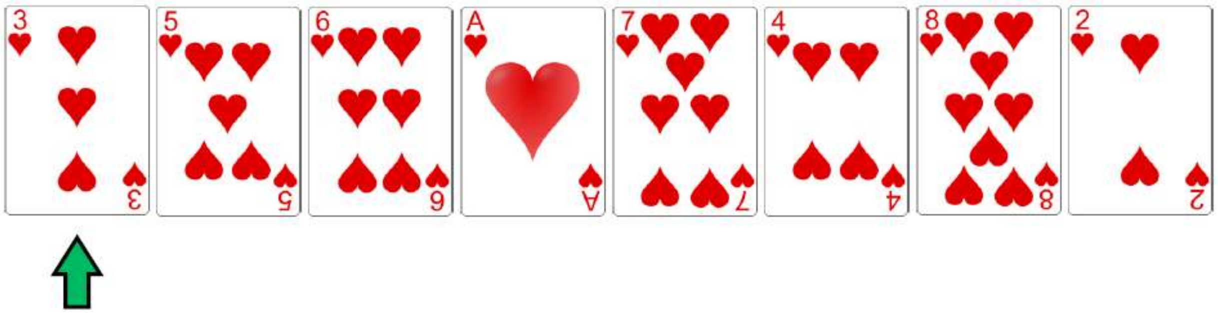
Insertion sort

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



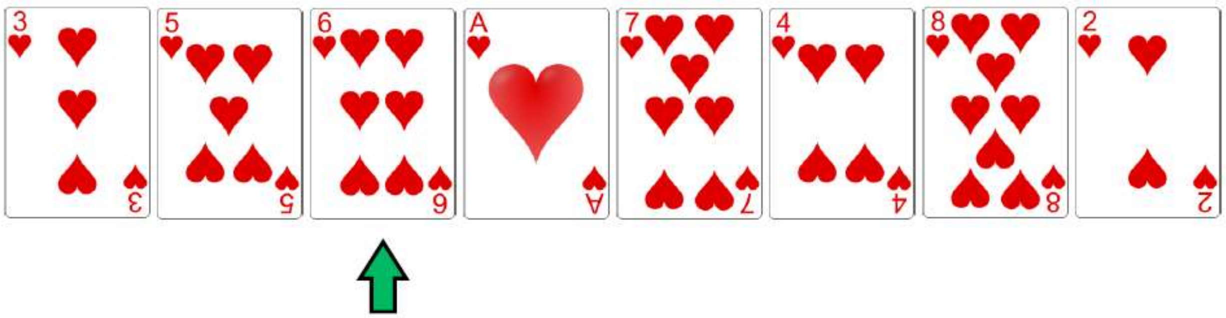
Insertion sort

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



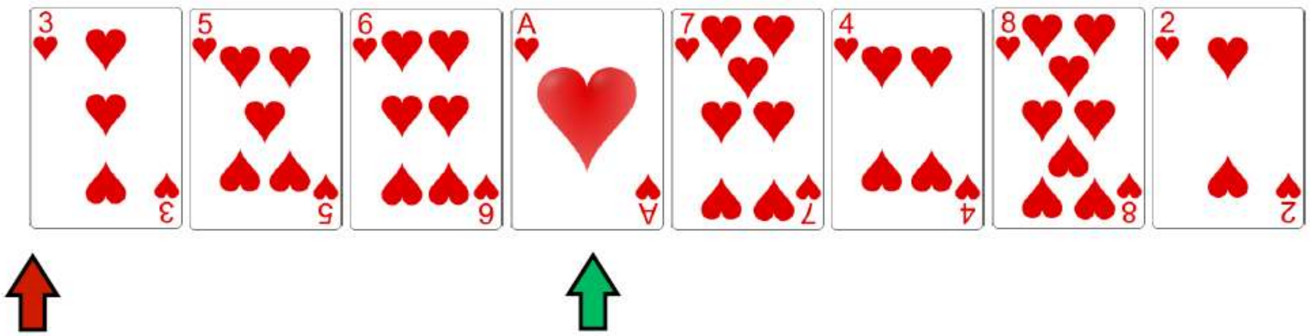
Insertion sort

- The third card (#6) is larger than the first two cards
- So, we don't need to move it



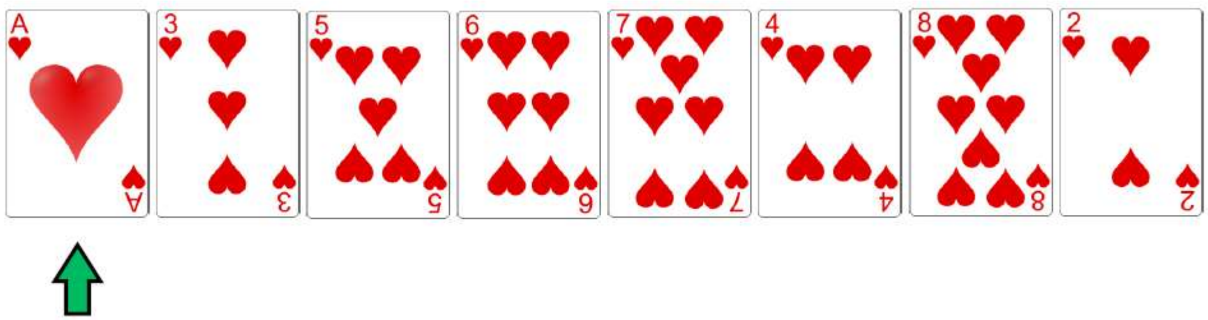
Insertion sort

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



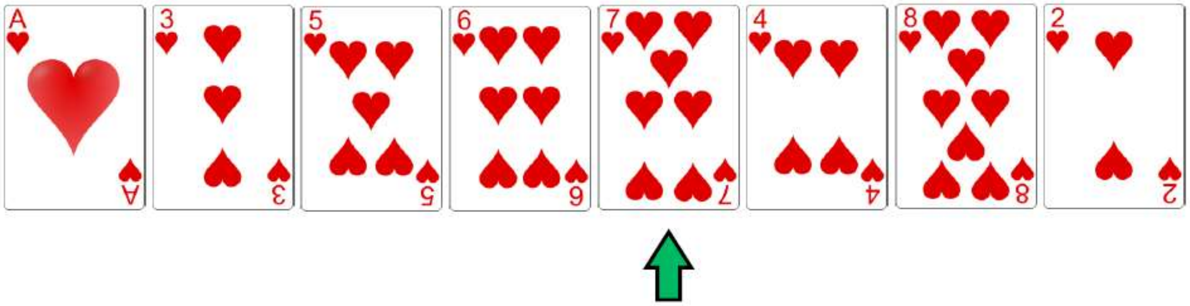
Insertion sort

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



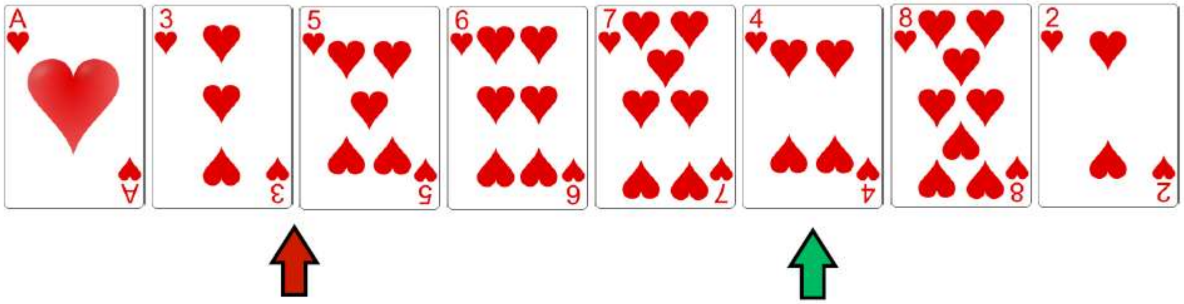
Insertion sort

- The fifth card (#7) is larger than the first four cards
- So, we don't need to move it



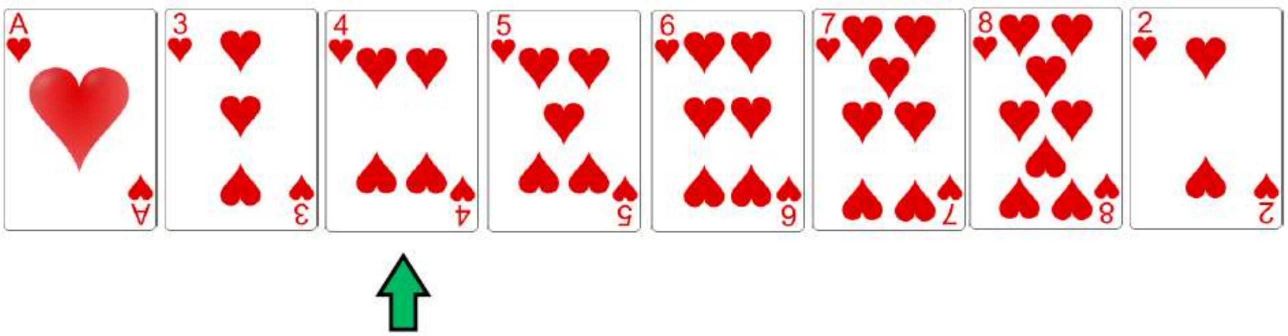
Insertion sort

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



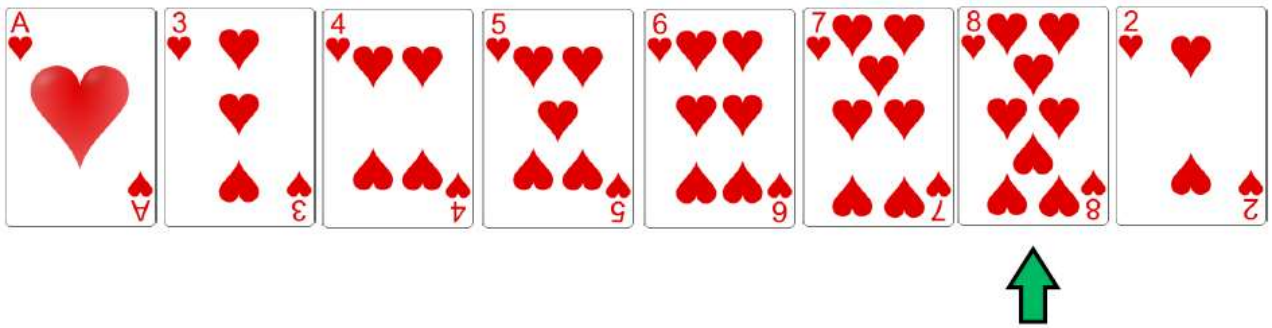
Insertion sort

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



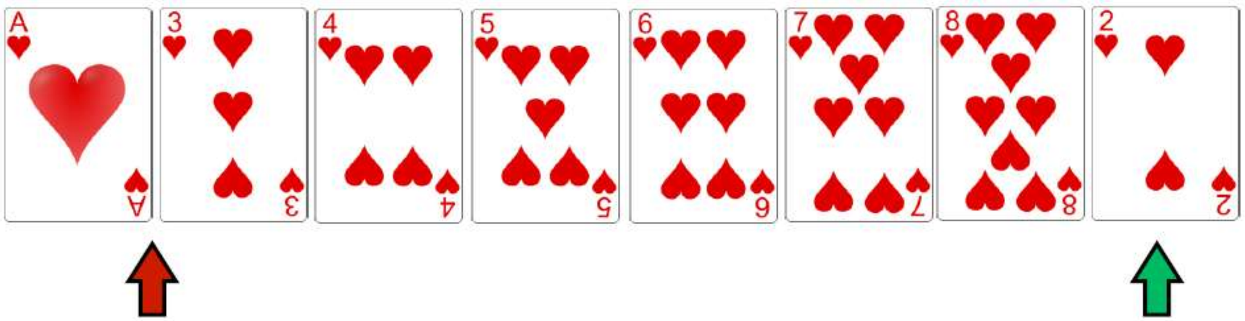
Insertion sort

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



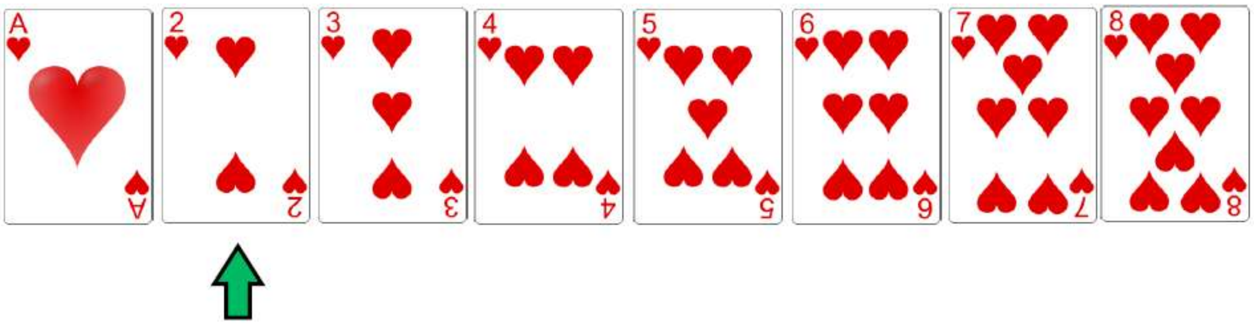
Insertion sort

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



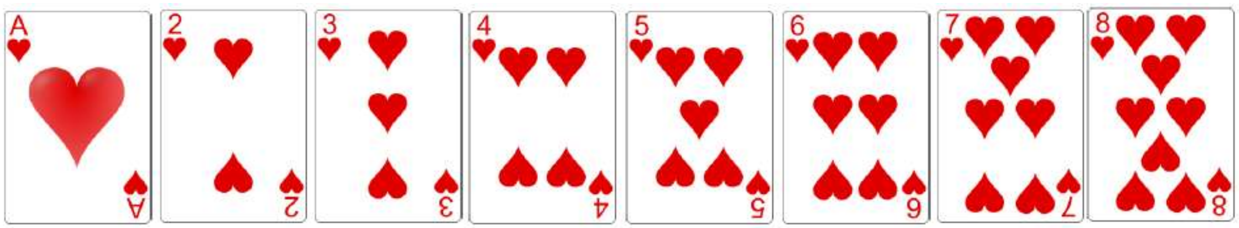
Insertion sort

- 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 have just confirmed there can be different ways to solve the same computational problem
  - ➔ can derive many different **algorithms** for solving the sorting problem
- Algorithm:
  - A set of concrete steps
  - Steps solve a problem or accomplish some task
  - Solve in a finite amount of time
  - The earliest algorithm known as Euclid's algorithm dates from 300 BC and used to find the Lowest Common Denominator of two numbers.
- The Selection Sort and Insertion Sort algorithms are only two ways of sorting a list of values
- New problem: Wish to sort a list of student records by their GPAs.
  - Would both of these algorithms work?
  - Yes! A hallmark of a good algorithm is that it is general ➔ can solve a wide variety of similar problems

Limits of Computation

What computer can do?	What 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 difference investment options based on historical data.	Choose a perfect investment or predict success and future of companies.
Find information about colleges offering computer science course.	Make a perfect decision on the best school to attend.
Solve well defined problems.	Solve ambiguous problems.

Unsolvable 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 1043 different games.

Computer solving one trillion combinations per second will compute the perfect game of chess if we are patient enough to wait 10^{21} years, so it is only unsolvable in a practical sense.

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 problem types
- 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 very peculiar field
  - it relies partly on old mathematical ideas
  - it advances in development of new techniques at an extraordinary pace
- The semester you will be exposed to many of the modern topics in CS and also to some of the older mathematical content that is still very relevant today

Questions?