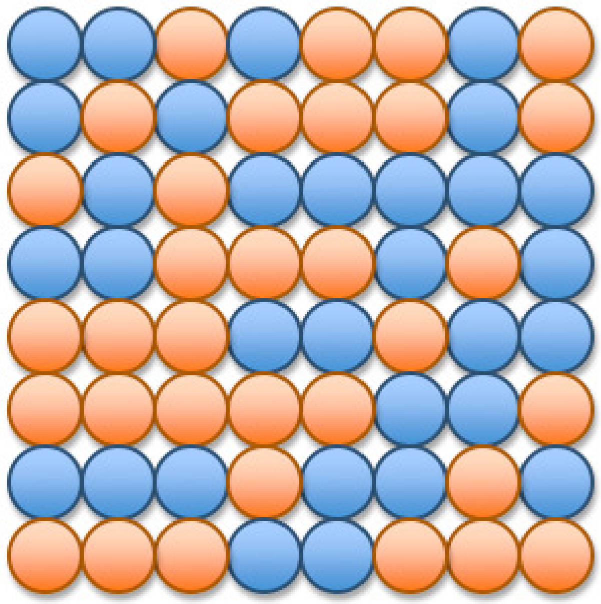
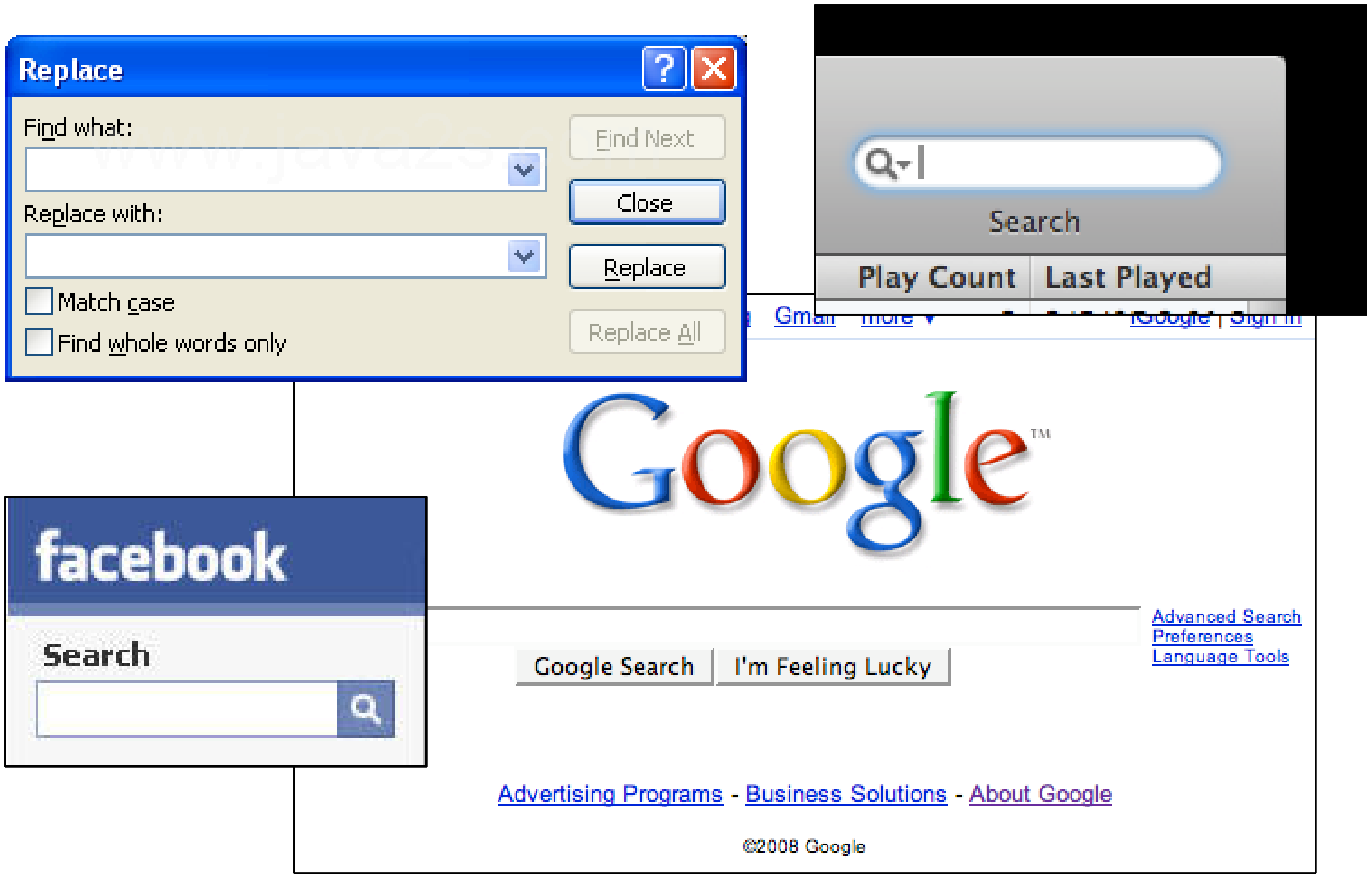


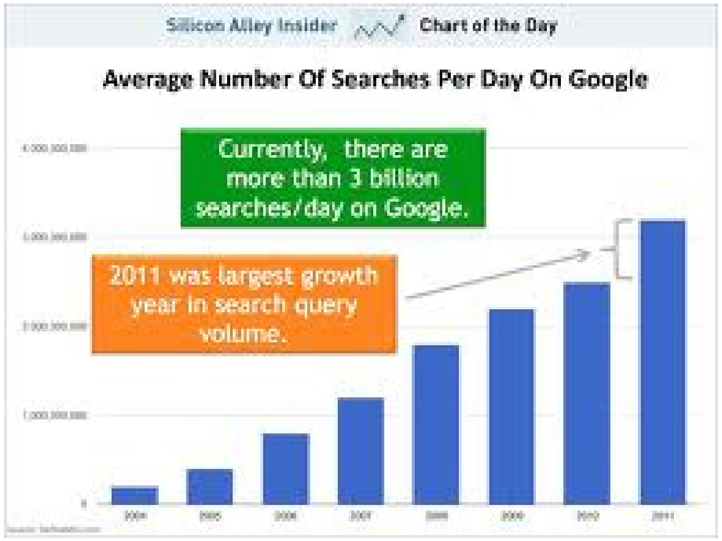
UNIT 4A
Iteration: Searching

Searching



Replace
Find what:
Replace with:
Match case
Find whole words only
Find Next
Close
Replace
Replace All
Search
Play Count
Last Played
Gmail
more ▾
iGoogle | Sign in
Google™
Google Search
I'm Feeling Lucky
Advanced Search
Preferences
Language Tools
Advertising Programs - Business Solutions - About Google
©2008 Google
facebook
Search

Searching



Twitter – 1.6 billion queries per day !!!!

Businessinsider.com

- **Google: 34,000 searches per second** (2 million per minute; 121 million per hour; 3 billion per day; 88 billion per month, figures rounded)
- **Yahoo: 3,200 searches per second** (194,000 per minute; 12 million per hour; 280 million per day; 8.4 billion per month, figures rounded)
- **Bing: 927 searches per second** (56,000 per minute; 3 million per hour; 80 million per day; 2.4 billion per month, figures rounded)

http://searchengineland.com

Goals of this Unit

- Study an iterative algorithm called linear search that finds the first occurrence of a target in a collection of data.
- Study an iterative algorithm called insertion sort that sorts a collection of data into non-decreasing order.
- Learn how these algorithm scale as the size of the collection grows.
- Express the amount of work each algorithm performs as a function of the amount of data being processed.

Review

Nested Loops revisited..

- Nested for example

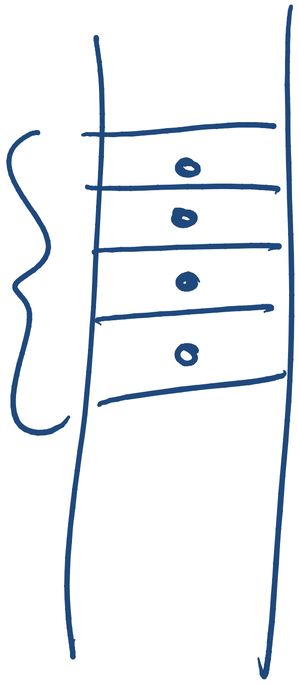
Ruby times method

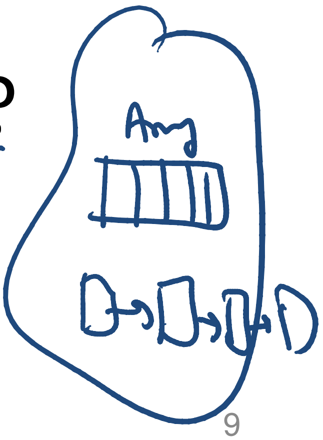
- Syntax: some_integer.times {statements}
- Some_integer.upto(i)
- Some_integer.downto(i)

Containers

Type of Containers

- A set is an unordered container
- A list is an ordered container
  (has an index)
- An array is a list
  homogeneous, contiguous
- The difference between arrays and lists?





Ruby Arrays

Empty Array A = []

A.length

Examples:

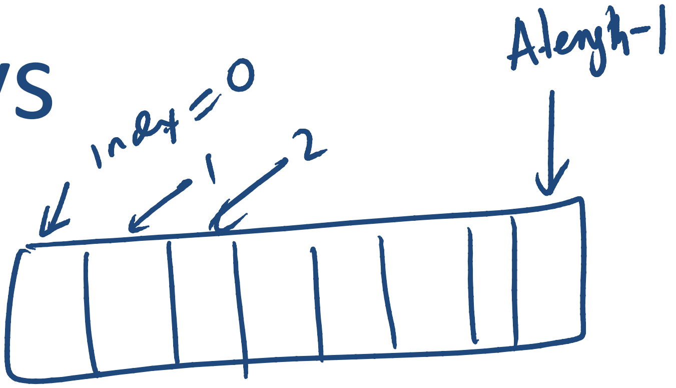
Create: A= (1..9).to_a() (from a range)

A = [1,2,3,4,5,6,7,8,9] (as a list)

Access: A[0] (first) A[A.length-1] (last)

A[i] (any index)

define



Why Study Containers?

- Organizing data for processing
- Example
  - Music player
    - How to store music?
      - pop ~~My~~PlayList = ["call me baby", "blah blah", "yesterday"]
      - MyPlyList = [popPlyList, country, "JT"]
    - How to design the program?

      ✓
    - How to write the code?

      ✓

Searching

What can we search?

- Cats & dogs
- recepies
- pictures, movies, songs
- ~~[illegible]~~ facts
- Socks
-

Designing a simple search algorithm

- The problem: Given a list of items
  - Find if a specific item exists or not
  - Find if more than one of the specific items exists
  - Find the first item in the list
  - Find the last item in the list

Class Demo

From demo to program

- How to store data?

  As a list with index

- How to write a search function?

  ?

- Write code
  - Using basic loops
  - Using ruby methods

Ruby Array methods

- Suppose A is an array
  - A.include?(arg)
  - A.index?(arg)
  - A.length

Ruby Style search

```
movies = ["up", "wall-e", "toy story", "monsters inc",
            "cars", "bugs life",  "finding nemo",
            "the incredibles", "ratatouille"]

movies.include?("wall-e") =>
movies.include?("toy") =>
movies.index("cars") =>
movies.index("shrek") =>
movies.index("Up") =>
```

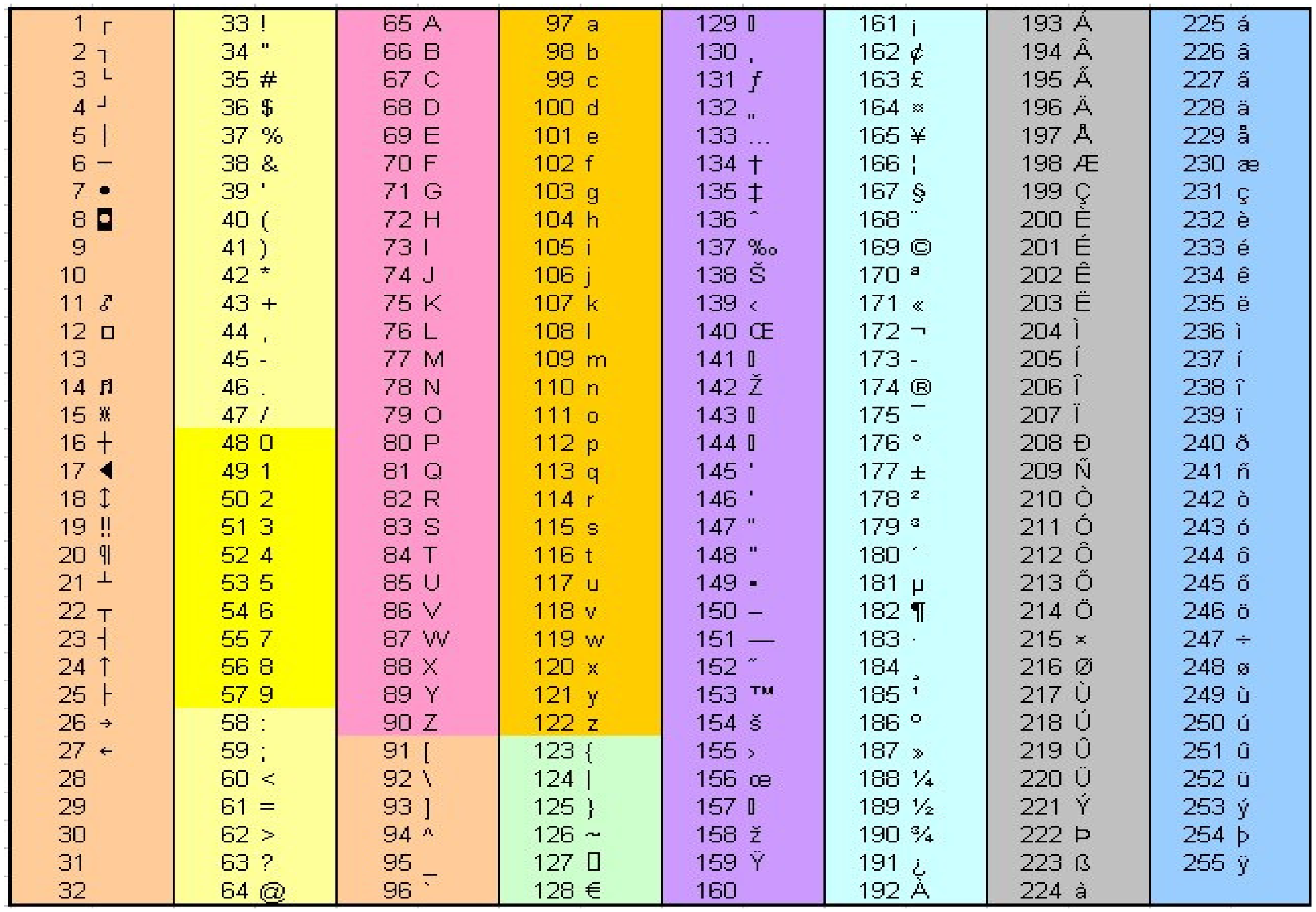
How strings are represented

A Little More about Strings

- You can use relational operators to compare strings: <, <=, >, >=, ==, !=

- Comparisons are done character by character using ASCII codes.

Extended ASCII table



1	┌	33	!	65	A	97	a	129	▯	161	¡	193	Á	225	á
2	┐	34	"	66	B	98	b	130	‚	162	¢	194	Â	226	â
3	└	35	#	67	C	99	c	131	ƒ	163	£	195	Ã	227	ã
4	┘	36	$	68	D	100	d	132	„	164	¤	196	Ä	228	ä
5	│	37	%	69	E	101	e	133	…	165	¥	197	Å	229	å
6	─	38	&	70	F	102	f	134	†	166	¦	198	Æ	230	æ
7	•	39	'	71	G	103	g	135	‡	167	§	199	Ç	231	ç
8	◘	40	(	72	H	104	h	136	ˆ	168	¨	200	È	232	è
9		41	)	73	I	105	i	137	‰	169	©	201	É	233	é
10		42	*	74	J	106	j	138	Š	170	ª	202	Ê	234	ê
11	♂	43	+	75	K	107	k	139	‹	171	«	203	Ë	235	ë
12	□	44	,	76	L	108	l	140	Œ	172	¬	204	Ì	236	ì
13		45	-	77	M	109	m	141	▯	173	-	205	Í	237	í
14	♫	46	.	78	N	110	n	142	Ž	174	®	206	Î	238	î
15	☼	47	/	79	O	111	o	143	▯	175	¯	207	Ï	239	ï
16	┼	48	0	80	P	112	p	144	▯	176	°	208	Ð	240	ð
17	◄	49	1	81	Q	113	q	145	'	177	±	209	Ñ	241	ñ
18	↕	50	2	82	R	114	r	146	'	178	²	210	Ò	242	ò
19	‼	51	3	83	S	115	s	147	"	179	³	211	Ó	243	ó
20	¶	52	4	84	T	116	t	148	"	180	´	212	Ô	244	ô
21	┴	53	5	85	U	117	u	149	▪	181	µ	213	Õ	245	õ
22	┬	54	6	86	V	118	v	150	–	182	¶	214	Ö	246	ö
23	┤	55	7	87	W	119	w	151	—	183	·	215	×	247	÷
24	↑	56	8	88	X	120	x	152	˜	184	¸	216	Ø	248	ø
25	├	57	9	89	Y	121	y	153	™	185	¹	217	Ù	249	ù
26	→	58	:	90	Z	122	z	154	š	186	º	218	Ú	250	ú
27	←	59	;	91	[	123	{	155	›	187	»	219	Û	251	û
28		60	<	92	\	124	|	156	œ	188	¼	220	Ü	252	ü
29		61	=	93	]	125	}	157	▯	189	½	221	Ý	253	ý
30		62	>	94	^	126	~	158	ž	190	¾	222	Þ	254	þ
31		63	?	95	_	127	▯	159	Ÿ	191	¿	223	ß	255	ÿ
32		64	@	96	`	128	€	160		192	À	224	à		

Exercise on String Comparison

"Steelers" > "Jets" =>

"steelers" > "Jets" =>

"Steelers" > "jets" =>

"Steeler Nation" > "Steelers" =>

" Steeler Nation" > "Steelers" =>

A contains? method

Find Last

Design an algorithm that returns the index of the last occurrence of a key in a list if the key is present, or **`nil`** otherwise.