Strings

Strings

- Our civilization has been built on them
  - but we are moving towards to digital media
- Anyway, handling digital media is similar to ..
- A string is a list of characters
  - 성균관대학교정보통신학부
  - Am I a string?
  - 一片花飛減却春

Character Code - ASCII

- Define 2**7 = 128 characters
  - why 7 bits?
- char in C is similar to an integer
  - ‘D’ + ‘a’ – ‘A’

0	NUL	1	SOH	2	STX	3	ETX	4	EOT	5	ENQ	6	ACK	7	BEL
8	BS	9	HT	10	NL	11	VT	12	NP	13	CR	14	SO	15	SI
16	DLE	17	DC1	18	DC2	19	DC3	20	DC4	21	NAK	22	SYN	23	ETB
24	CAN	25	EM	26	SUB	27	ESC	28	FS	29	GS	30	RS	31	US
32	SP	33	!	34	”	35	#	36	$	37	%	38	&	39	’
40	(	41	)	42	*	43	+	44	,	45	-	46	.	47	/
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;	60	<	61	=	62	>	63	?
64	@	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z	91	[	92	/	93	]	94	^	95	_
96	‘	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	{	124	—	125	}	126	~	127	DEL

more about ASCII

- What about other languages?
  - there are more than 100 scripts
  - some scripts have thousands characters
- Unicode
  - several standards
    - UTF16
    - UTF8 –variable length, leading zero means ASCII set.
- Programming Languages
  - C, C++ treat char as a byte
  - Java treat char as two bytes

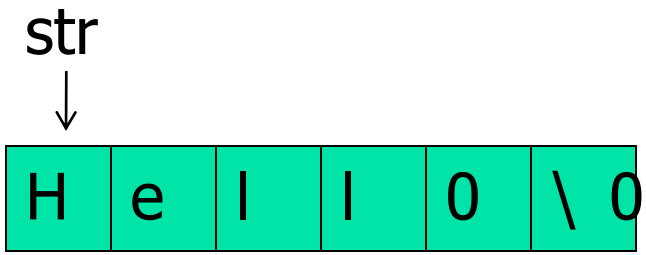
String Representations

- null-terminated: C, C++
- array with length: Java
- linked list of characters
  - used in some special areas

- Your Choice should consider
  - space occupied
  - constraints on the contents
  - O(1) access to i-th char (search, insert, delete)
  - what if the length can be limited (file names)

null-terminated

```
char str[6] =  “Hello” ;

char arr1[] =  “abc” ;
char arr2[] = { ‘a’ ,  ‘b’ ,  ‘c’ );
char arr3[] = { ‘a’ ,  ‘b’ ,  ‘c’ ,  ‘\0’ };
```



- need for the termincation

```
printf( “format string” , str);  /* str can be any length
```

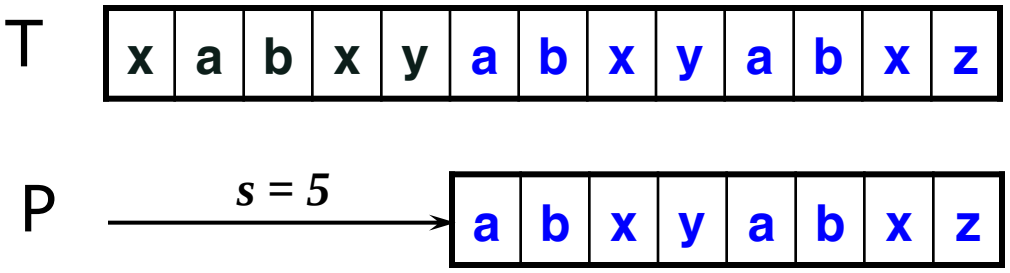
String Match

- Exact matching
  - a BIG problem for Google, twitter, NHN, ...
  - native method
  - preprocessing methods
  - Boyer-Moore Algorithm
- Inexact matching

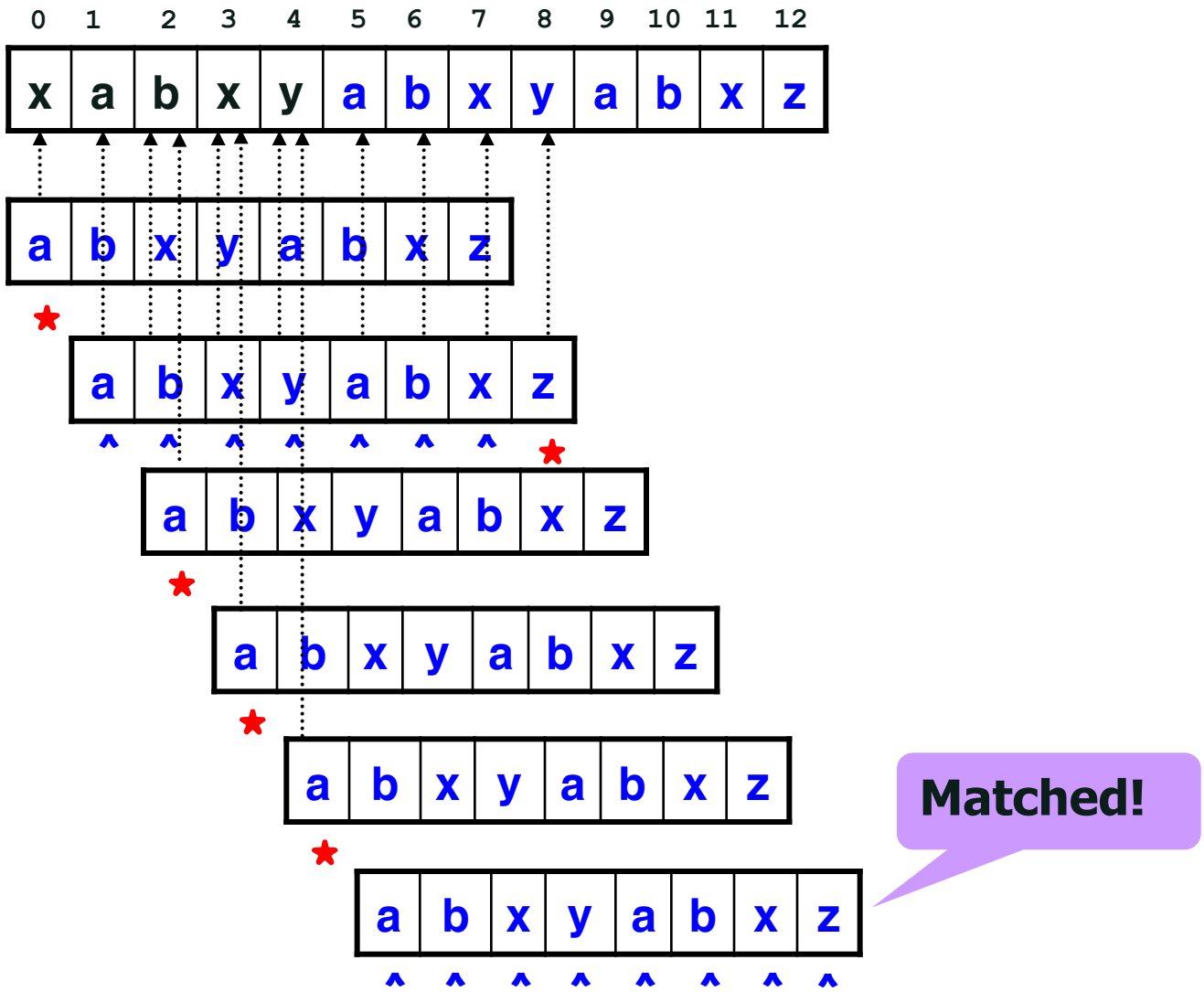
this problem itself can be a single whole course for a semester

String match(2)

- Problem
  - find P in T



- Your solution
  - how many comparisons?
  - better way?


0 1 2 3 4 5 6 7 8 9 10 11 12
x a b x y a b x y a b x z
a b x y a b x z
a b x y a b x z
a b x y a b x z
a b x y a b x z
a b x y a b x z
a b x y a b x z
Matched!

Problem Example

- There are millions documents that contain company names – Anderson, Enron, Lehman..
- M&A, bankrupt force them to be changed
- Your mission
  - change all the company names if they are changed

Input/Output Example

```
4
"Anderson Consulting" to "Accenture"
"Enron" to "Dynegy"
"DEC" to "Compaq"
"TWA" to "American"
5
Anderson Accounting begat Anderson Consulting, which
offered advice to Enron  before it DECLARED bankruptcy,
which made Anderson
Consulting quite happy it changed its name
in the first place!
```

Your Plan

- read the M&A list into a DB
  - define DB structure
  - how to handle the double quote (“) sign?
- main loop
  - compare each line of a doc with each DB entry
  - if there is a match, replace it with a new name
- Now, define
  - functions
  - global variables

Your naive solution

- read data for changed company names to build a DB

old name	new name
Anderson Consulting	Accenture
Enron	Dynegy
DEC	Compaq
....	.....

- for each entry
  - look for the whole documents
  - if there is a match, change it

```
#include <string.h>

#define MAXLEN          1001    /* longest possible string */
#define MAXCHANGES      101     /* maximum number of name changes */

typedef char string[MAXLEN];

string mergers[MAXCHANGES][2];  /* store before/after corporate names */
int nmergers;                   /* number of different name changes */
```

mergers[101][2]

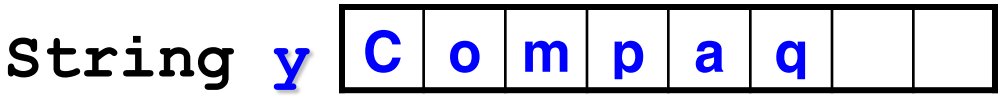
	0	1
0	Anderson Consulting\0	Accenture\0
1	Enron\0	Dynegy\0
2	DEC\0	Compaq\0
3	TWA\0	American\0

```
read_changes()
{
        int i;                  /* counter */

        scanf("%d\n",&nmergers);
        for (i=0; i<nmergers; i++) {
                read_quoted_string(&(mergers[i][0]));
                read_quoted_string(&(mergers[i][1]));
        }
}
```

```
read_quoted_string(char *s)
{
        int i=0;                /* counter */
        char c;                 /* latest character */

        while ((c=getchar()) != '\"') ;
        while ((c=getchar()) != '\"') {
                s[i] = c;
                i = i+1;
        }
        s[i] = '\0';
}
```

```
main()
{
        string s;                   /* input string */
        char c;                     /* input character */
        int nlines;                 /* number of lines in text */
        int i,j;                    /* counters */
        int pos;                    /* position of pattern in string */

        read_changes();
        scanf("%d\n",&nlines);
        for (i=1; i<=nlines; i=i+1) {          /* read text line */
                j=0;
                while ((c=getchar()) != '\n') {
                        s[j] = c;
                        j = j+1;
                }
                s[j] = '\0';

                for (j=0; j<nmergers; j=j+1)
                        while ((pos=findmatch(mergers[j][0],s)) != -1) {
                           replace_x_with_y(s, pos,
                                strlen(mergers[j][0]), mergers[j][1]);
                        }

                printf("%s\n",s);
        }
}
```

```
replace_x_with_y(char *s, int pos, int xlen, char *y)
{
        int i;                          /* counter */
        int slen,ylen;                  /* lengths of relevant strings */

        slen = strlen(s);
        ylen = strlen(y);

        if (xlen >= ylen)
                for (i=(pos+xlen); i<=slen; i++) s[i+(ylen-xlen)] = s[i];
        else
                for (i=slen; i>=(pos+xlen); i--) s[i+(ylen-xlen)] = s[i];

        for (i=0; i<ylen; i++) s[pos+i] = y[i];
}
```



Example Review

- Is it safe?
  - What if a quote sign is missing?
  - what if there is no room for a new long company name?
  - ....
- Is it efficient
  - space
  - time

String Library

```
#include <ctype.h>      /* include the character library */

int isalpha(int c);     /* true if c is either upper or lower case */
int isupper(int c);     /* true if c is upper case */
int islower(int c);     /* true if c is lower case */
int isdigit(int c);     /* true if c is a numerical digit (0-9) */
int ispunct(int c);     /* true if c is a punctuation symbol */
int isxdigit(int c);    /* true if c is a hexadecimal digit (0-9,A-F) */
int isprint(int c);     /* true if c is any printable character */

int toupper(int c);     /* convert c to upper case -- no error checking */
int tolower(int c);     /* convert c to lower case -- no error checking */
```

String Library 2

```
#include <string.h>      /* include the string library */

char *strcat(char *dst, const char *src);     /* concatenation */
int strcmp(const char *s1, const char *s2);   /* is s1 == s2? */
char *strcpy(char *dst, const char *src);     /* copy src to dist */
size_t strlen(const char *s);                 /* length of string */
char *strstr(const char *s1, const char *s2); /* search for s2 in s1 */
char *strtok(char *s1, const char *s2);       /* iterate words in s1 */
```

Doublets

- a dictionary up to 25,143 words, not exceeding 16 letters

Sample Input

```
booster
rooster
roaster
coasted
roasted
coastal
postal

booster roasted
coastal postal
```

Sample Output

```
booster
rooster
roaster
roasted

No solution.
```