CS 31 Discussion

ABDULLAH-AL-ZUBAER IMRAN

WEEK 6: ARRAYS AND STRINGS

Discussion Objectives

Review and practice things covered during lectures

- More on Arrays
- Strings
- C Strings
- Coding examples
- Project4

Programming Challenge

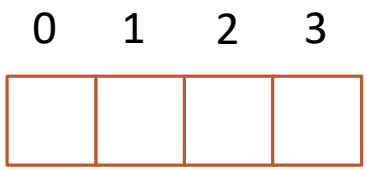
Time for you to ask questions!

Array

Declare an array

- <type> <name>[size]

```
int a[4];
```



`a[i]` is an `i`-th variable in the array `a`.

`size` should must be a positive integer constant.

```
int a[4];
```

```
const int N = 10;
int a[N];
```

You can treat each element of the array as a variable.

```
x[3] = 5;
x[1]++;
cout << x[i] << endl;
```

Initialization of an Array

```
int a[5] = {1, 2, 3, 5, 7};
int a[] = {1, 2, 3, 5, 7};
```

You cannot set the size to what's less than the number of elements in the initialization statement.

However, it is okay to set the size to what's more than the number of elements in the initialization statement.

```
int a[3] = {1, 2, 3, 5, 7}; // (right/wrong?)
int a[10] = {1, 2, 3, 5, 7}; // (right/wrong?)
```

Common mistakes

```
int a[10];
for (int i = 0; i < a.size(); i++) {
 ...
}
```

No `size()` function is defined for arrays.

```
const int SIZE = 10;
int a[SIZE];
for (int i = 0; i < SIZE; i++) {
 ...
}
```

Common mistakes

Out-of-range access not allowed!

```
int a[10];
a[15] = 5; // error
a[-10] = 4; // error
a[9] = 10; // okay
```

Arrays in a Function

```
void fibo10(int fib[]);
```

Note that the size of fib is not specified, you can explicitly pass the size in the function.

```
void fibo10(int fib[], int n);
```

Arrays in a Function

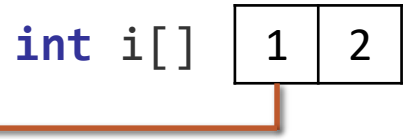
Question: Will the code compile? If so, what's the output?

```
#include <iostream>
#include <string>

using namespace std;
int foo (int a[]);
int main () {
  int i[] = {1, 2};
  foo(i);
  cout << i[0] << endl;
}
int foo (int a[]) {
  return a[0]++;
}
```

Output:
2

Pass by Pointer



C Strings

Why we learn C strings?

An array of characters

C strings are **null-terminated**

- Every C string ends with a marker called the **zero byte** (**null character**), which we use an escape sequence `\0` to represent (its ASCII number is 0).

The maximal length of this string is 9 (not 10)

```
char s[10];
```

Note that for `string`, there is no null terminator.

C Strings

To initialize a string

```
char s[10] = "HOWAREYOU";
```

H	O	W	A	R	E	Y	O	U	\0

```
char s[] = "HOWAREYOU";
```

```
char s[10] = “hello";
```

h	e	l	l	o	\0				

```
char s[10] = “Hello, world!";
```

C Strings

Question: Will this code compile? What's the output?

```
#include <iostream>
using namespace std;

int main() {
  char s[10] = "HOWAREYOU";
  s[3] = '\0';
  cout << s << endl;
}
```

H	O	W	A	R	E	Y	O	U	\0

↓

H	O	W	\0	R	E	Y	O	U	\0

Output:
HOW

C Strings

Question: Will this code compile? What's the output?

```
#include <iostream>
using namespace std;

int main() {
  char s[10];
  s = "abcdefg";
  cout << s << endl;
}
```

Output:

Cannot compile.
Direct assignment like that works only in an initialization expression.

C Strings

Question: Will this code compile? What's the output?

```
#include <iostream>
#include <string>

using namespace std;
int main () {
  char a[4] = "ca\0";
  cout << a << endl;
}
```

Output:
ca

c	a	\0	\0

C Strings

Question: Will this code compile? What's the output?

```
#include <iostream>
#include <string>
using namespace std;

int main() {
  string s = "hello";
  cout << s.size() << ' ';
  s[1] = '\0';
  cout << s.size() << '\n';
}
```

Output:

```
5 5
```

Functions for C Strings

```
#include <cstring>
```

Operation	What it does
`strlen(s)`	Returns the length of `s`, not counting `'\0'`.
`strcpy(t,s)`	Copies the string `s` to `t`. (Notes: `t=s` won't do the job. Also, `strcpy` doesn't do the size check for you. You must make sure there's enough space in t yourself.)
`strncpy(t,s,n)`	Copies the first n characters of `s` to `t`. (Note: No size check.)
`strcat(t,s)`	Appends `s` to the end of `t`. (Notes: No size check. `t += s` won't do the job.)
`strcmp(t,s)`	Compares `t` and `s`. Returns `0` if they are equal, something greater than `0` if `t > s`, and something less than `0` if `t < s`. (Note: `t == s` or `t < s` won't do the job.)

C Strings

Two alternatives to traverse a C string.

```
char s[10] = "HOWAREYOU";
for (int k = 0; t[k] != '\0'; k++)
  cout << t[k] << end;
```

```
#include<cstring>
...

char s[10] = "HOWAREYOU";
for (int k = 0; k < strlen(s); k++)
  cout << t[k] << end;
```

C Strings

Convert a C string into a C++ string, and vice versa.

```
char cs[10] = "hello";
string cpps;
cpps = cs;

string cpps = cs;
string cpps(cs);
```

C string to string

```
char cs[10];
string cpps = "hello";
strcpy(cs, cpps.c_str());
```

string to c string

You cannot:

```
char cs[10] = cpps.c_str();
```

```
char cs[10];
cs = cpps.c_str();
```

C Strings

Question: Will this code compile? What's the output?

```
#include <iostream>
#include <cstring>

using namespace std;
int main () {
  char a[] = "sup";
  char b[] = "hey\0you";
  char c[] = {'\0‘};
  cout << strlen(a) << endl;
  cout << strlen(b) << endl;
  cout << strlen(c) << endl;
}
```

Output:

```
3
3
0
```

C Strings

Question: Will this code compile? What's the output?

```
#include <iostream>
#include <cstring>

using namespace std;
int main () {
  char a[]= "sup";
  char A[] = "SUP";

  cout << strcmp(a, A) << endl;
  cout << strcmp(A, a) << endl;
}
```

Output:

32

-32

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

ASCII table

C Strings

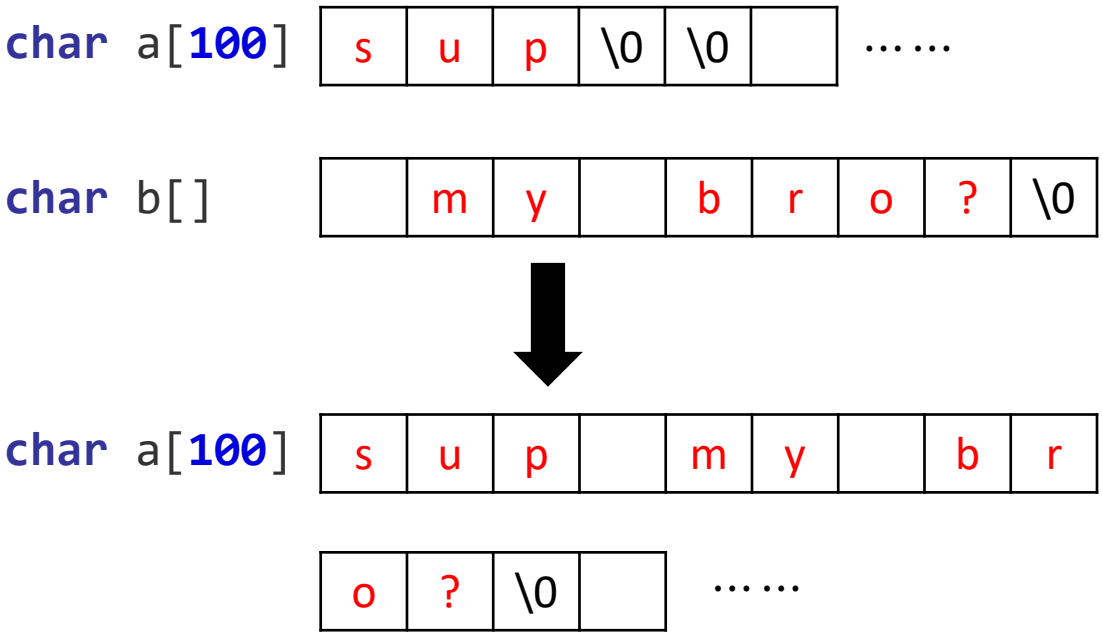
Question: Will this code compile? What's the output?

```
#include <iostream>
#include <cstring>

using namespace std;
int main () {
  char a[100] = "sup\0";
  char b[] = " my bro?";

  strcat(a, b);
  cout << a << endl;
}
```

Output:
sup my bro?



Project 4

int countMatches(const string a[], int n, string target) →*Return the number of strings in the array that are equal to target.*

int detectMatch(const string a[], int n, string target) →*Return the position of a string in the array that is equal to target.*

bool detectSequence(const string a[], int n, string target, int& begin, int& end) →*Find the earliest occurrence in a of one or more consecutive strings that are equal to target.*

int detectMin(const string a[], int n) →*Return the position of a string in the array such that that string is <= every string in the array.*

int moveToBack(string a[], int n, int pos) →*Eliminate the item at position pos by copying all elements after it one place to the left.*

int moveToFront(string a[], int n, int pos) →*Eliminate the item at position pos by copying all elements before it one place to the right.*

Project4

int detectDifference(const string a1[], int n1, const string a2[], int n2) →Return the position of the first corresponding elements of a1 and a2 that are not equal.

int deleteDups(string a[], int n)→ For every sequence of consecutive identical items in a, retain only one item of that sequence.

bool contains(const string a1[], int n1, const string a2[], int n2)→If all n2 elements of a2 appear in a1, in the same order (though not necessarily consecutively), then return true.

int meld(const string a1[], int n1, const string a2[], int n2, string result[], int max)→If a1 has n1 elements in nondecreasing order, and a2 has n2 elements in nondecreasing order, place in result all the elements of a1 and a2, arranged in nondecreasing order, and return the number of elements so placed.

int split(string a[], int n, string splitter)→Rearrange the elements of the array so that all the elements whose value is < splitter come before all the other elements, and all the elements whose value is > splitter come after all the other elements. Return the position of the first element that, after the rearrangement, is not < splitter, or n if there are no such elements.

Project4

Pay attention to the return values for each of the functions (especially, when you need return -1)

For each of the functions moveToBack, moveToFront, deleteDups, meld, and split, if the function is correctly implemented, you will earn one bonus point for that function if it does its job without creating any additional array.

Your program must not use any function templates from the algorithms portion of the Standard C++ library. If you don't know what the previous sentence is talking about, you have nothing to worry about.

Your implementations must not use any global variables whose values may be changed during execution.

Your program must build successfully under both g31 and either Visual C++ or clang++.

Thanks!

Questions?

Some of the materials presented have been taken from other TA discussions