STRING

MAN

http://teacher.buet.ac.bd/ali_nayeem/CSE109_Feb2015/

CT3 ON ARRAY, STRING

- Next Next Sunday 17/5/15
- Materials:
  - 2 slides
  - Only relevant sections from
    - TUC: Ch 5
    - LUC: Ch 8,9

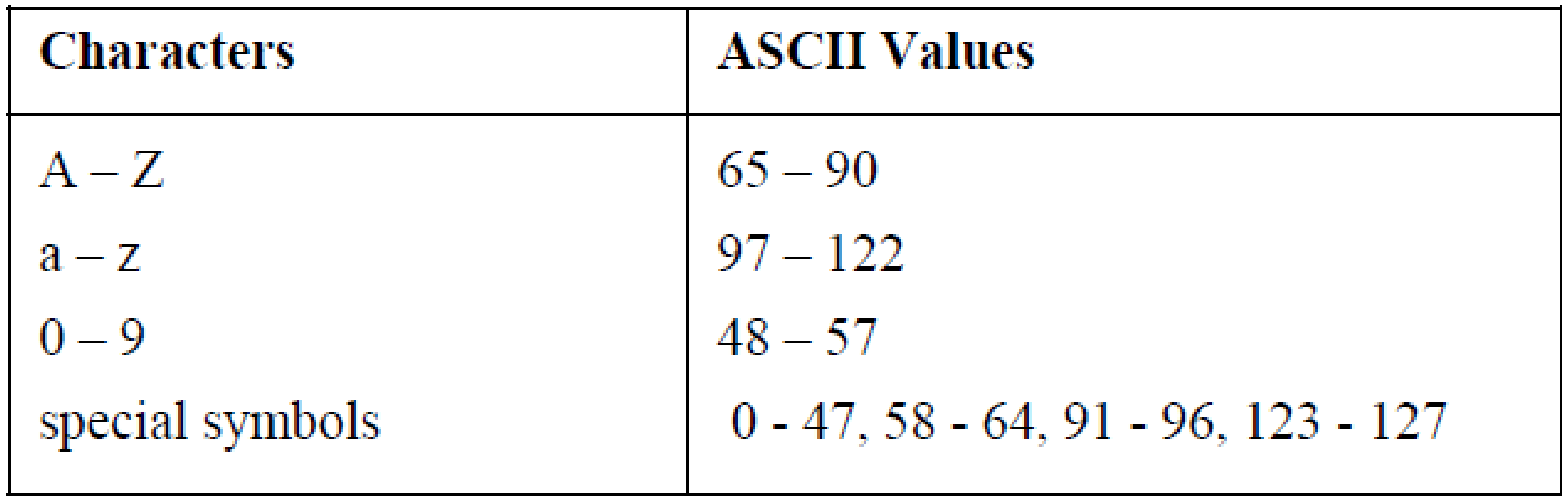
RECAP: CHARACTER IN C

Character Constants

- Size 8 bits
- Like small integer (0-255)
- Rules for constructing character constant
  - a single alphabet, a single digit or a single special symbol enclosed within single inverted commas.
  - The maximum length of a character constant can be 1 character.
- Example
  - 'A'
  - 'I'
  - '5'
  - '='

How character is stored in memory

- Needs represent character by integer
- Needs a standard
  - American Standard Code for Information Interchange (ASCII)



Characters	ASCII Values
A – Z	65 – 90
a – z	97 – 122
0 – 9	48 – 57
special symbols	0 - 47, 58 - 64, 91 - 96, 123 - 127

ASCII

Value	Char	Value	Char	Value	Char	Value	Char	Value	Char	Value	Char
0		22	▬	44	,	66	B	88	X	110	n
1	☺	23	↨	45	-	67	C	89	Y	111	o
2	☻	24	↑	46	.	68	D	90	Z	112	p
3	♥	25	↓	47	/	69	E	91	[	113	q
4	♦	26	→	48	0	70	F	92	\	114	r
5	♣	27	←	49	1	71	G	93	]	115	s
6	♠	28	∟	50	2	72	H	94	^	116	t
7	•	29	↔	51	3	73	I	95	_	117	u
8	◘	30	▲	52	4	74	J	96	`	118	v
9	○	31	▼	53	5	75	K	97	a	119	w
10	◙	32		54	6	76	L	98	b	120	x
11	♂	33	!	55	7	77	M	99	c	121	y
12	♀	34	"	56	8	78	N	100	d	122	z
13	♪	35	#	57	9	79	O	101	e	123	{
14	♫	36	$	58	:	80	P	102	f	124	|
15	☼	37	%	59	;	81	Q	103	g	125	}
16	►	38	&	60	<	82	R	104	h	126	~
17	◄	39	'	61	=	83	S	105	i	127	^{M}H
18	↕	40	(	62	>	84	T	106	j	128	Ç
19	‼	41	)	63	?	85	U	107	k	129	ü
20	¶	42	*	64	@	86	V	108	l	130	é
21	§	43	+	65	A	87	W	109	m	131	â

ASCII

Value	Char	Value	Char	Value	Char	Value	Char	Value	Char	Value	Char
132	ä	154	Ü	176	░	198	╞	220	▄	242	≥
133	à	155	¢	177	▒	199	╟	221	▌	243	<
134	å	156	£	178	▓	200	╚	222	▐	244	⌠
135	c	157	¥	179	│	201	╔	223	▀	245	⌡
136	ê	158	Pts	180	┤	202	╩	224	α	246	÷
137	ë	159	ƒ	181	╡	203	╦	225	β	247	≈
138	è	160	á	182	╢	204	╠	226	Γ	248	°
139	ï	161	í	183	╖	205	═	227	π	249	•
140	î	162	ó	184	╕	206	╬	228	Σ	250	·
141	ì	163	ú	185	╣	207	╧	229	σ	251	√
142	Ä	164	ñ	186	║	208	╨	230	µ	252	η
143	Å	165	Ñ	187	╗	209	╤	231	τ	253	²
144	É	166	ª	188	╝	210	╥	232	Φ	254	■
145	æ	167	º	189	╜	211	╙	233	θ	255	
146	Æ	168	¿	190	╛	212	╘	234	Ω		
147	ô	169	⌐	191	┐	213	╒	235	δ		
148	ö	170	¬	192	└	214	╓	236	∞		
149	ò	171	½	193	┴	215	╫	237	ø		
150	û	172	¼	194	┬	216	╪	238	Є		
151	ù	173	¡	195	├	217	┘	239	∩		
152	ÿ	174	«	196	─	218	┌	240	≡		
153	Ö	175	»	197	┼	219	█	241	±		

Character Variable

- Type char
- Format specifier %c for printf and scanf

```
char a, b, d ;
a = 'F' ;
b = 'G' ;
d = '+' ;
```

- ASCII values of the characters are stored in the variables.

Arithmetic Operators

- +
- -
- *
- /
- %

INPUT CHARACTERS

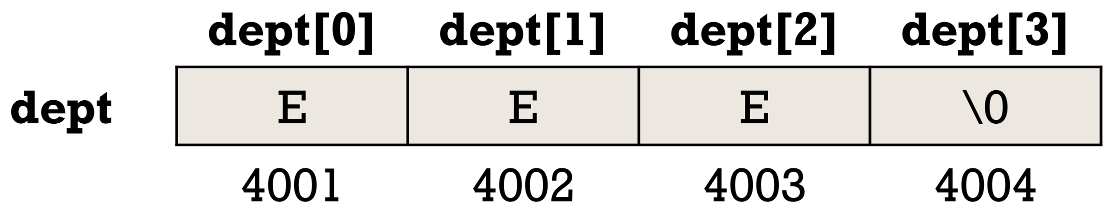
- getche()/getch()/getchar/scanf() can be used
- getchar()
  - Compiler dependent behaviour
  - Waits until carriage return
  - Read only one char
  - Other input and carriage return will be in buffer
  - Subsequent input (e.g, scanf) will consume them
  - Can cause trouble
  - Defined in stdio.h
- getche()/getch()
  - Return immediately after a key is pressed
  - Defined in conio.h

STRING

- Most common use of one dimensional array is string
- One dimensional character array terminated by a null (‘\0’)
- ‘\0’ & ‘0’ are not same
- ASCII Value of ‘\0’ is 0
- ASCII Value of ‘0’ is 48
- Array size must be at least one byte larger than the string size to make room for the null
- Terminating null is important
  - Indicates where string ends
- A string constant is automatically null-terminated by the compiler

STRING

- `char dept[]={‘E’, ‘E’, ‘E’, ‘\0’};`
- `char dept[]=“EEE”; /*string constant*/`
  - Shortcut for initializing string
  - ‘\0’ is not necessary in this declaration



STRING

```
#include<stdio.h>

int main()
{
        char course[]="CSE109";
        int i=0;
        while(course[i])
        {
                printf("%c\n", course[i]);
                i++;
        }
        return 0;
}
```

Output:

```
C
S
E
1
0
9
```

STRING

- Null
  - False
  - Value 0

STRING READ

- %s
  - In scanf
  - Reads characters untill `ENTER` pressed
  - `ENTER` key is not stored, replaced with null character
  - No bound checking
  - Can not read multi word string separated by space
    - "Department Name: EEE"
  - scanf("%s", s);

STRING READ

- gets()
  - Library function
  - Defined in stdio.h
  - Call it using the name of the character array without using index
    - gets(s)
  - Reads characters untill `ENTER` pressed
  - `ENTER` key is not stored, replaced with null character
  - No bound checking
  - Can receive multiword string

STRING WRITE

- %s in printf
  - printf(“%s”, s);
- puts()
  - puts(“hello”)
  - puts(s)

MEMORY LAYOUT

- Each character occupies one byte of memory

	dept[0]	**dept[1]**	**dept[2]**	**dept[3]**
dept	‘E’	‘E’	‘E’	‘\0’
	4001	4002	4003	4004

- Number of possible values in a string of length 3 is 255^3

CAN YOU FIND?

- The length of a string
- Check whether a string is palindrome?
  - abcdcba
  - acca
- Lowercase/uppercase
- Frequency of each character in a string
  - aabegggfdd
    - a:2
    - b:1
    - d:1
    - e:1
    - f:1
    - g:3

USES OF STRINGS

- Working with very large integers
  - Suppose you are given a large integer. The integer may have up to 128 decimal digits. You have to detect whether the integer is divisible by 3 or not.
  - **Hint**: A number is divisible by 3 if the **sum of its digits** is divisible by 3.

STRING LIBRARY FUNCTIONS

- Required to include `string.h`
- Strlen() : Finds the length of the string
- strcat(to, from): : Appends one string at the end of the other
- strcpy(to, from): Copies one string into another
- strcmp (s1, s2): Compares two strings
  - Returns 0 if same
  - -ve if s1 less than s2
  - +ve if s1 greater than s2

STRING LIBRARY FUNCTIONS

- strcpy

```
char str1[]="CSE110";
char str2[]="CSE109";
for(int i=0; str2[i]!=‘\0’,;i++)
  str1[i]=str2[i];
str1[i]=‘\0’;
```

CAN YOU SOLVE IT?

- Take as input the names of N students and find how many names contain the word “nayeem”

STRING TABLES

- 2D array of characters
- Arrays of strings
  - Each element is a string
- char names[10][40]
  - 10 names (strings) each can hold 40 characters at most including null
- gets(names[2]);
- printf(names[1]);

INITIALIZE

```
char names[][10]={
                  “Joy”,
                  “Fahad”,
                  “Alamin”,
                  “Noman”,
                  “Tareq”
                  };
```

1001	J	o	y	\0						
1011	F	a	h	a	d	\0				
1021	A	l	a	m	i	n	\0			
1031	N	o	m	a	n	\0				
1041	T	a	r	e	q	\0				

1050 (last location)

THANKS TO

- Johra Muhammad Moosa
  - Lecturer
  - Department of Computer Science & Engineering
  - Bangladesh University of Engineering & Technology