Introduction to C Programming

— Characters and Strings —

Waseda University

Today's Topics

- Characters and Strings
  - ASCII (American Standard Code for Information Interchange)
  - Type of char
  - Strings (special character "¥0" or "\0")
  - Declaration and assignments
  - Strings as parameters of a function
- Standard library functions for strings
  - strcpy
  - strlen
  - strcmp

How to deal with characters

- To recognize characters in computers, C encodes a specified character into a binary integer. Most modern character-encoding schemes are based on ASCII.

- The form of declaration for English alphabets 'A' and 'a' is:

```
char x, y;      /* Declare the type char variables */
x='A';          /* Assign 'A' to x (0x41 in the ASCII chart) */
y='a';          /* Assign 'a' to x (0x61 in the ASCII chart) */
```

- Characters are enclosed in single quotes (ex. 'a').

ASCII

- ASCII is a character-encoding scheme.
- ASCII codes represent text in computers.
- The following is the ASCII chart:

	0x00	0x10	0x20	0x30	0x40	0x50	0x60	0x70
0x00	¥0 (NUL)	(control character)	(space)	0	@	P	‘	p
0x01	(control character)	(control character)	!	1	A	Q	a	q
0x02	(control character)	(control character)	”	2	B	R	b	r
0x03	(control character)	(control character)	#	3	C	S	c	s
0x04	(control character)	(control character)	$	4	D	T	d	t
0x05	(control character)	(control character)	%	5	E	U	e	u
0x06	(control character)	(control character)	&	6	F	V	f	v
0x07	(control character)	(control character)	’	7	G	W	g	w
0x08	(control character)	(control character)	(	8	H	X	h	x
0x09	¥ t (tab)	(control character)	)	9	I	Y	i	y
0x0a	¥ n (linefeed)	(control character)	*	:	J	Z	j	z
0x0b	(control character)	(control character)	+	;	K	[	k	{
0x0c	(control character)	(control character)	,	<	L	¥	l	—
0x0d	¥ r (carriage return)	(control character)	-	=	M	]	m	}
0x0e	(control character)	(control character)	.	>	N	^	n	~
0x0f	(control character)	(control character)	/	?	O	_	o	(control character)

Characters

- The type char represents single characters.
- The variable of the type char contains one character.
- Characters are enclosed in single quotes (ex. ‘a’).

```
char x;          /* Declare the type char variable */
x='A';           /* Assign ‘A’ to x (0x41 in the ASCII chart) */
```

- Functions ‘printf’ and ‘scanf’ can recognize characters as well as integers or floating point numbers.

```
char x;
scanf("%c",&x);  /* Assign a character from keyboard to x */
printf("%c",x);  /* Print the content of x on the screen */
```

Strings

- Strings are just arrays of character.

```
char s[100];                    /* Arrays for 100 characters */
```

- The special character '¥0' or '\0' (NUL) is used to indicate the end of a string.

```
char x[100];
x[0] = 'I'; x[1] = 'T'; x[2] = 'b'; x[3] = '¥0';
```

- We have to allocate one character for the end-of-string marker.
- Functions 'printf' and 'scanf' can recognize strings.

```
scanf("%s",x);    /* Assign a string to character array x */
printf("%s",x);              /* Print strings on the screen */
```

- The operator & is not necessary.
- Any string up to 99 characters long can be stored in x.

Example

Input a string and print it on the screen

```
#include <stdio.h>
int main(void){
  char Name[100];  /* Arrays for 100 characters (limited to 99 characters) */
  printf("Input your name:");
  scanf("%s",Name);                    /* Input a string (&Name is illegal) */
  printf("Your name is %s.¥n",Name);
  return 0;
}
```

- If "WASEDA" is inputed, each element of Name is

[0]	[1]	[2]	[3]	[4]	[5]	[6]
'W'	'A'	'S'	'E'	'D'	'A'	'¥0'

- The printf function prints characters just before '¥0'.
- The scanf function cannot recognize the space character. The string is separated into two different strings at that time.

Initializing strings

- If no dimension of an array is given, C will determine the dimension from the number of elements in the initialization list.

```
int S[]= {1, 2, 3, 4, 5};
```

- Strings can be initialized in a similar manner.

```
char Name[]={'W','A','S','E','D','A',' ','T','a','r','o','¥0'};
```

(The NUL character '¥0' is necessary at the end of characters.)

- C has a special shorthand for initializing strings:

```
char Name[]= "WASEDA Taro";      /* Special shorthand */
```

(Surround the string with double quotes.)

Exercise

Write a program (name.c) as follows:

- Input your family and last name (Declare two arrays of char)
- Print your name on the screen
- The output should be as follows:

```
Input family name: WASEDA
Input given name: Taro
Your name is WASEDA Taro.
```

Strings as parameters of function

Use pointers for inputing strings to a function

```
#include <stdio.h>
void message(char *mes){
   printf("Message: %s ¥n",mes);          /* Print the inputted string */
}
int main(void){
   char x[] = "program started.";         /* Arrays for characters */
   char *y  = "program is running.";      /* Pointer for a string */
   message(x);                            /* Input array */
   message(y);                            /* Input pointer */
   message("program ended.");             /* Input String */
   return 0;
}
```

The output will be

```
Message: program started.
Message: program is running.
Message: program ended.
```

Standard functions for strings

To use standard functions for strings, string.h is necessary.

strcpy: Copy strings

- strcpy(str1, str2): Copy str2 into str1

```
char Name[] = "WASEDA Taro";     /* Arrays for characters */
Name[0] = 'T'; Name[2]='K';      /* Change each element */
Name = "OHKUMA Jiro";            /* This is illegal. */
strcpy(Name,"OHKUMA Jiro");      /* The strcpy function copies strings. */
```

- This function executes the following operation:

```
void mystrcpy (char *x, char *y){
    int i;
    for (i=0; ;i++){              /* no condition */
        x[i] = y[i];
        if(y[i]=='¥0') break;     /* When NUL appears, this loop ends.*/
    }
}
```

Standard functions for strings

strlen: Length of characters

- strlen(str): Outputs length of characters
- The NUL character is not counted.

```
char Name[100];
scanf("%s",x);                  /* Input string */
printf("Name length is %d.¥n", strlen(Name));   /* Print length of string */
```

strcmp: Compare two strings

- strcmp(str1, str2): Returns 0 if str1 equals str2, otherwise nonzero.

```
char x[100], y[100];
scanf("%s",x);                  /* Input string */
scanf("%s",y);                  /* Input string */
if(strcmp(x,y)==0) printf("Same names. ¥n);
```

- When we write "if (x==y)", it compares the addresses of these arrays. Then it doesn't works well.

Summary

- Characters and Strings
  - ASCII (American Standard Code for Information Interchange)
  - Type of char
  - Strings (special character “¥0” or “\0”)
  - Declaration and assignments
  - Strings as parameters of a function
- Standard library functions for strings
  - strcpy
  - strlen
  - strcmp