Data input and output

Single character input, single character output, scanf, printf, puts, gets, functions, interactive programming.

INPUT/OUTPUT FUNCTIONS

- There are numerous library functions available for I/O, through which data can be read from or written to files or standard I/O devices.
- These library functions can be classified into three broad categories:
  a) *Console I/O functions* - functions to receive input from keyboard and write output to VDU.
  b) *Disk I/O functions* - functions to perform I/O operations on a floppy disk or hard disk.
  c) *Port I/O functions* - functions to perform I/O operations on various ports.

Console Input/Output functions

Formatted Functions			Unformatted Functions		
Type	**Input**	**Output**	**Type**	**Input**	**Output**
char	scanf()	printf()	char	getch()	putch()
				getchar()	putchar
				getche()	
int	scanf()	printf()	int	-	-
float	scanf()	printf()	float	-	-
string	scanf()	printf()	string	gets()	puts()

Console Input/Output functions

- The basic difference between formatted and unformatted I/O functions is that the

  formatted functions allow the

  input read from the keyboard or

  the output displayed on the VDU

  to be formatted as per our requirements.

- The two functions used for this purpose are **printf()** and **scanf()**

Formatted Console I/O Functions

- **scanf( )** allows us to enter data from the keyboard that will be formatted in a certain way.

The general form of scanf( ) statement is as follows:

scanf(control string, arg1, arg2.....argn);

- control string refers to a string containing certain required formatting information,
- arg1, arg2,....argn are arguments that represent the individual data items.

scanf("%d %f %c",&c,&a,&ch);

Formatted Console I/O Functions

- **printf()** translates internal values to character.

printf(control string, arg1, arg2, ……..argn)

The format string can contain:

- Characters that are simply printed as they are.
- Conversion specification that begins with a % sign.
- Escape sequences that begins with a \ sign.
- Printf() converts, formats, and prints its arguments on the standard output under the control of the format.
- It returns the number of characters printed.

Formatted Console I/O Functions

- The format string contains two types of objects
  - ordinary characters, which are copied to the output stream,
  - and conversion specifications,
- each of which causes conversion and printing of the next successive argument to printf().
- Each conversion specification begins with % and ends with a conversion character.

Formatted Console I/O Functions

- If the character after the % is not a conversion specification, the behavior is undefined.

Data type		Conversion character
Integer	short signed	%d or %I
	short unsigned	%u
	long signed	%ld
	long unsigned	%lu
	unsigned hexadecimal	%x
	unsigned octal	%o
Real	float	%f
	double	%lf
Characters	signed char	%c
	unsigned char	%c
string		%s

Unformatted Console I/O Functions

- There are several standard library function available under this category.
- These functions deals with a single character or with a string of characters.
- The functions that can handle one character at a time.
- **a) Single Character Input - the *getchar* Function**
- **b) Single Character Output - the *putchar* Function**
- **c) String Input and String Output Function**

Unformatted Console I/O Functions

Single Character Input - the *getchar* Function

- This function reads one character from the keyboard after the new-line character is received (when press Enter key).
- The function does not require any arguments, though a pair of empty parentheses must follow the word *getchar*.
- In general terms, getchar function is written as

character variable = getchar();

- Where *character variable* refers to some previously declared character variable.

```
char ch;
ch = getchar ( ) ;
```

Unformatted Console I/O Functions

Single Character Output - the *putchar* Function :

- *putchar()*, the opposite of getchar(), is used to put exactly one character on the screen.
- Putchar requires as an argument the character to put on the screen.
- In general the putchar function is written as

  putchar(character variable);
- where *character variable* refers to some previously declared character variable

```
char ch;
ch = getchar ( ) ; /* input a character from kbd*/
putchar (ch) ;     /* display it on the screen */
```

Unformatted Console I/O Functions

String Input and String Output Function :

gets( ) - The gets( ) function receives a string from the keyboard.

- The scanf( ) function has some limitations while receiving a string of characters.
- The moment a blank character is typed, scanf( ) assumes that the end of the data is being entered. So it is possible to enter only one word string using scanf( ).
- To enter multiple words in to the string, the gets( ) function can be used.
- In general terms, gets function is written as

gets(variable name);

- Where *variable name* will be a previously declared variable.

Unformatted Console I/O Functions

- **puts( ) -** The *puts( )* function works exactly opposite to *gets( )* function.
- It outputs a string to the screen. Puts( ) can output a single string at a time.
- In general terms, gets function is written as

  puts(variable name);
- Where *variable name* will be a previously declared variable.