DATA INPUT OUTPUT

For inputting and outputting data we use library function .the important of these functions are getch( ), putchar( ), scanf( ), printf( ), gets( ), puts( ). For using these functions in a C-program there should be a preprocessor statement #include<stdio.h>.

[A preprocessor statement is a statement before the main program, which begins with # symbol..] .

stdio.h is a header file that contains the built in program of these standard inputoutput
function.

getchar function

It is used to read a single character (char type) from keyboard. The syntax is
char variable name = getchar( );

Example:
char c;
c = getchar( );

For reading an array of characters or a string we can use getchar( ) function.

Example:

```
#include<stdio.h>
main( )
{
char place[80];
int i;
for(i = 0;( place [i] = getchar( ))! = '\n', ++i);
}
```

This program reads a line of text.

putchar function

It is used to display single character. The syntax is
putchar(char c);

Example:

```
            char c;
            c = 'a';
            putchar(c);
```

Using these two functions, we can write a very basic program to copy the input, a character at a time, to the output:

```
#include <stdio.h>

/* copy input to output */

main()
{
        int c;

        c = getchar();

        while(c != EOF)
                {
                putchar(c);
                c = getchar();
                }

        return 0;
}
```

scanf function

This function is generally used to read any data type- int, char, double, float, string.
The syntax is

scanf (control string, list of arguments);

The control string consists of group of characters, each group beginning % sign and a conversion character indicating the data type of the data item. The conversion characters are c,d,e,f,o,s,u,x indicating the type resp. char decimal integer, floating point value in exponent form, floating point value with decimal point, octal integer, string, unsigned integer, hexadecimal integer. ie, “%s”, “%d” etc are such group of characters.

An example of reading a data:

```
         #include<stdio.h>
                       main( )
                       {
          char name[30], line;
          int x;
          float y;
          .........
          ..........
          scanf("%s%d%f", name, &x, &y);
          scanf("%c",line);
         }
```

NOTE: 1) In the list of arguments, every argument is followed by & (ampersand symbol) except
string variable.

2) s-type conversion applied to a string is terminated by a blank space character. So string having blank space like “Govt. Victoria College” cannot be read in this manner. For reading such a string constant we use the conversion string as “%[^\n]” in place of “%s”.

Example:

```
char place[80];
…………….
scanf("%[^\n]", place);
……………..
```

with these statements a line of text (until carriage return) can be input the variable ‘place’.

printf function

This is the most commonly used function for outputting a data of any type.
The syntax is

printf(control string, list of arguments)

Here also control string consists of group of characters, each group having % symbol and
conversion characters like c, d, o, f, x etc.

Example:

```
`                #include<stdio.h>
                   main()
                   {
                   int x;
                   scanf("%d",&x);
                   x*=x;
                   printf("The square of the number is %d",x);
                   }
```

Note that in this list of arguments the variable names are without &symbol unlike in the
case of scanf( ) function. In the conversion string one can include the message to be displayed. In the above example "The square of the number is" is displayed and is followed by the value of x.For writing a line of text (which include blank spaces) the
conversion string is "%s" unlike in scanf function. (There it is "[^\n]").

More about printf statement

There are quite a number of format specifiers for `printf`. Here are the basic ones :

```
        %d      print an int argument in decimal
        %ld     print a long int argument in decimal
        %c      print a character
        %s      print a string
        %f      print a float or double argument
        %e      same as %f, but use exponential notation
        %g      use %e or %f, whichever is better
        %o      print an int argument in octal (base 8)
        %x      print an int argument in hexadecimal (base 16)
        %%      print a single %
```

To illustrate with a few more examples: the call

```
        printf("%c %d %f %e %s %d%%\n", '1', 2, 3.14, 56000000.,
"eight", 9);
```

would print

```
        1 2 3.140000 5.600000e+07 eight 9%
```

The call

```
        printf("%d %o %x\n", 100, 100, 100);
```

would print

```
        100 144 64
```

Successive calls to `printf` just build up the output a piece at a time, so the calls

```
        printf("Hello, ");
        printf("world!\n");
```

would also print `Hello, world!` (on one line of output).

While inputting or outputting data field width can also be specified.This is included in the conversion string.(if we want to display a floating point number convert to 3 decimal places the conversion string is “%.3f”).For assigning field width,width is placed before the conversion character like “%10f”,”%8d”,”%12e” and so on…Also we can display data making correct to a fixed no of decimal places.
For example if we want to display x=30.2356 as 30.24 specification may be “%5.2f” or simply “%.2f”.