CHAPTER 3

INPUT AND OUTPUT HANDLING

SHORT AND LONG QUESTIONS

Q1. Define input functions.

Ans: Input Functions:

In computer programming input means to feed data into a program through an input device. C language provides the **scanf(), getch(), getche(), getchar()** and **gets()** functions to input data.

Q2. Define output functions.

Ans: Output Functions:

In computer programming, output means to display information on screen or print on printer. C language provides the printf(), putchar() and puts() functions to output information on computer screen.

Q3. What is single character output function? Explain with examples.

Ans: Displaying Single Characters

The Putchar() Function:

This function is sued to output a single character to the screen.

Syntax:

Putchar(character);

Examples 1:

Putchar('A')

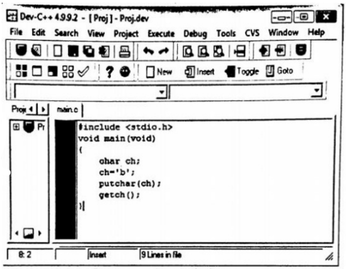
Examples 2:

Char m= 'P'

Putchar(m)

Program:

The program in Fig prints the character 'b' stored in variable ch, on the screen using the putchar() function.


Dev-C++ 4.9.9.2 - [Proj] - Proj.dev
File Edit Search View Project Execute Debug Tools CVS Window Help
New Insert Toggle Goto
Proj
main.c
Pr
8: 2
Insert
9 Lines in file

```
#include <stdio.h>
void main(void)
{
    char ch;
    ch='b';
    putchar(ch);
    getch();
}
```

Program to print a single character

The first statement in the main() function declares the variable ch as type char. Second statement assigns it the letter 'b' and the Putchar() function prints it on the screen.

Q4. What is displaying string output function? Explain with example.

Ans: Displaying Strings:

A sequence of characters is known as a string. Strings are very commonly used in programming languages for storing words, name, addresses, sentences etc. String are always enclosed in double quotes as "Computer"

The put() Functions:

The put() function is sued to print a string on the screen.

Syntax:

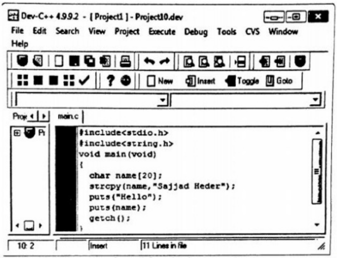
Puts(string);

Examples:

Putchar("AssalamoAllaikum");

Program:

The program in Fig, demonstrates the use of puts() function.


Dev-C++ 4.9.9.2 - [Project1] - Project10.dev
File Edit Search View Project Execute Debug Tools CVS Window Help
New Insert Toggle Goto
Proj
main.c
10: 2
Insert
11 Lines in file

```
#include<stdio.h>
#include<string.h>
void main(void)
{
  char name[20];
  strcpy(name,"Sajjad Heder");
  puts("Hello");
  puts(name);
  getch();
}
```

Use of puts() function in a program

The first statement of the program declares the string variable name and the second statement stores "SajjadHeder" in it. The **strcpy()** function is used for this purpose. The **strcpy()** function stores the string on the right side in the quotation marks to the variable on the left side. The third statement print the text **"Hello"** and the last statement prints the text stored in the variable name.

The **puts()** function is a special-purpose output function. It can only output one string at a time. It is used when the programmer wants to output a single string. It is required to include the **string.h** header file at the beginning of the program when this function is used in a program.

Q5. Explain the displaying of text, variables, constants and expressions using printf() function in C language?

Ans: Displaying of text, variables, constants and expression:

The printf() Function:

The **printf()** function is used to print text and values on the screen in a specified format. It is also called **formatted output function.**

Syntax:

The general syntax of this function is:

Printf(control_string, list of arguments);

The **control_string** consists of text, the formal specifiers and the escape sequence. It is written within double quotes. Text specifies the message that is to be printed along with the values. Format specifier specifies the format according to which a value is to be printed. Escape sequence controls the printing on the output device.

List of arguments consists of a list of variables or arithmetic expression, separated by commas, whose values are to be printed. The values are printed according to the corresponding format specifier. The first format specifier applies to the first argument, the second to the second argument and so on. The arguments in the **printf()** function are optional. When **printf()** function is used to print only a message, then arguments are omitted.

Example:

Following Is an example of **printf()** function.

Printf("\n The value of a is %d and the value of b is %d,", a, b);

Assuming that a has the value of 3 and b as the value of 4, then the following message will be printed by the above statement on a new line.

The value of a is 3 and the value of b is 4.

In the above print statement, \n is an escape sequence and %d is a format specifier. The escape sequence \n is used to start printing on a new line. The

format specifier %d is replaced by the values stored in variables a and b in the order in which they are written. The format specifier %d is used for printing values stored in integer variables.

Q6. Briefly explain the scanf() function with the help of program.

Ans: The scanf() Function:

The scanf() function is used to get values into variables from the keyboard during the execution of a program. The value is input into a variable in a specified format.

Its **syntax** is: **scanf(control_string, list of variables):**

The **control_string** in **scanf()** specifies the format specifiers. It is written within double quotes. The **control_string** in **scanf()** function is different from **printf()** function. In **scanf()** function strings cannot be given.

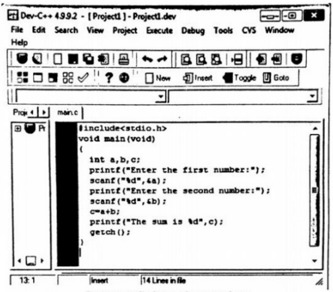
List of variables consists of a list of variables, separated by comma into which the values are to be entered. In C language, **&** symbol is used in **scanf()** function before the name of the variable to which a value is to be assigned.

For example, to input values into two integer variables, **a** and **b,** the **scanf()** function is written as:

Scanf("%d,%d", &a, &b):

Here is the format specifier %d is used twice in control string for the variables a and b. When this statement is executed, the values may be entered by separating them by comma and the user has to press the Enter key after entering the data.

The program in Fig reads two numbers and print their sum.



Dev-C++ 4.9.9.2 - [Project1] - Project1.dev
File Edit Search View Project Execute Debug Tools CVS Window Help
New Insert Toggle Goto
Proj
main.c
13: 1
Insert
14 Lines in file

```
#include<stdio.h>
void main(void)
{
  int a,b,c;
  printf("Enter the first number:");
  scanf("%d",&a);
  printf("Enter the second number:");
  scanf("%d",&b);
  c=a+b;
  printf("The sum is %d",c);
  getch();
}
```

Program to find sum of two numbers

In the above program, the statement

Inta, b,c;

Is used to declare that **a, b** and **c** are integer variables. Note that the statement ends with a semicolon () as all the C statements do.

The statement

Printf("Enter the first number:");

Prompts the user for the first number

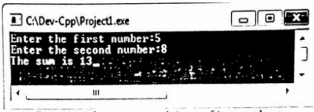
The statement

Scanf("%d, &a);

Reads the value for the variable a and stores it in the computer's memory. In this statement, **%d** format specifier specifies that an integer is to be input.

The next two statements prompt the user to input the second number and store it in variable named b. The expression c=a+b calculates the sum of values stored in variables a and b and stores it in variable c. The last statement print the sum on the screen.

The execution of this program is shown Fig.



```
Enter the first number:5
Enter the second number:8
The sum is 13_
```

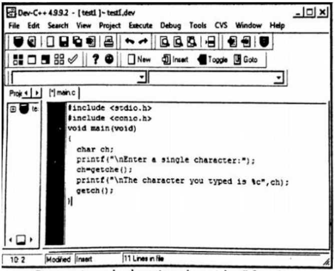
Execution of program to print sum of two numbers

Q7. Describe the getchar() function.

Ans: The getchar() Function:

This input function is used to read a single character from the keyboard. When this function Is used, we have to press the Enter key after typing the character.

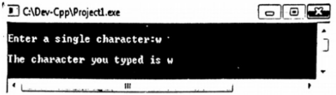
The program in Fig, reads a character using **getchar()** function and displays it on the screen.


Dev-C++ 4.9.9.2 - [test1] - test1.dev
File Edit Search View Project Execute Debug Tools CVS Window Help
New Insert Toggle Goto
Proj
[*] main.c
10: 2
Modified
Insert
11 Lines in file

```
#include <stdio.h>
#include <conio.h>
void main(void)
{
  char ch;
  printf("\nEnter a single character:");
  ch=getche();
  printf("\nThe character you typed is %c",ch);
  getch();
}
```

Program to read a character using getchar() function

The execution of this program is shown in Fig.



```
Enter a single character:w
The character you typed is w
```

Readingacharacterusinggetchar() function

Q8. What is the statement terminator in C?

OR

Write down the uses of statement terminator in C.

Ans: Statement Terminator:

Semicolon (;) is entered at the end of a statement in C language. It indicates the compiler that the statement ends here. If a statement is not terminated with semicolon, C compiler will give an error message during compilation and the program will not be compiled.

Q9. What are the operators of C Language? List the types of operators.

Ans: operators of C Language:

Expressions consist of constants and variables combined together with operators. An operator is a symbol used to command the computer to perform a certain mathematical or logical operation. Operators are used to operate on data and variables.

The following types of operators are commonly used in C language.

- Arithmetic operators
- Assignment operators
- Relational operators
- Logical operators
- Increment and decrement operators

Q10. Define arithmetic operators and its types. Also explain how to use them in a program.

Ans: Arithmetic Operators:

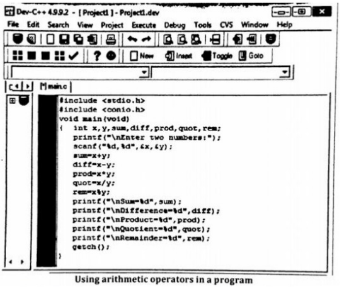
Arithmetic operators are used to perform arithmetic operations that include addition, subtraction, multiplication, division and also to find the remainder obtained when an integer is divided by another integer.

Types of Arithmetic Operators:

The types of arithmetic operators used in C programming are described with their operations in table.

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Remainder (Modulus) Operator

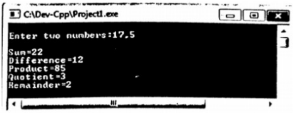
The program in Fig, demonstrates the use of arithmetic operators.


Dev-C++ 4.9.9.2 - [Project1] - Project1.dev
File Edit Search View Project Execute Debug Tools CVS Window Help
New Insert Toggle Goto
[*] main.c

```
#include <stdio.h>
#include <conio.h>
void main(void)
{   int x,y,sum,diff,prod,quot,rem;
    printf("\nEnter two numbers:");
    scanf("%d,%d",&x,&y);
    sum=x+y;
    diff=x-y;
    prod=x*y;
    quot=x/y;
    rem=x%y;
    printf("\nSum=%d",sum);
    printf("\nDifference=%d",diff);
    printf("\nProduct=%d",prod);
    printf("\nQuotient=%d",quot);
    printf("\nRemainder=%d",rem);
    getch();
}
```

Using arithmetic operators in a program

In this program when x/y is performed it will give an integer result because the fractional part is truncated when two operands are of type integer. Moreover, the remainder operator will give the remainder after dividing x by y when x%y is performed. The output of the program is shown in Fig.



```
Enter two numbers:17,5

Sum=22
Difference=12
Product=85
Quotient=3
Remainder=2
```

Use of arithmetic operators to perform calculations

Q11. Explain the use of increment and decrement operators in C with the help of examples.

Ans: Increment and Decrement Operators:

Increment operator is ++ and decrement operator Is --. These are defined in table.

Operator	Definition
++	Increment by 1
--	Decrement by 1

Examples:

++n and n++ are both equivalent to **n=n + 1 (or n+ 1)**

--n and n-- are both equivalent to **n=n – 1 (or n-=1)**

When increment or decrement operator is written before the variable, it is known as prefix and when it is written after the variable, it is known as postfix.

In certain situations, ++n and n++ have different effect. This is because ++n increments n before using its value whereas n++ increments n after it Is used.

As an example, suppose n has the value 3

The statement: **a= ++n;**

Will first increment **n** and then assigns the value 4 to a.

But the statement **a= n++;**

Will first assign the value 3 to a and then increments n to 4. In both cases n has the value 4.

The same rule applies to **–n** and **n—**as well

Q12. Differentiate between assignment operator and equal to operator?

Ans: Difference between Assignment Operator and Equal to Operator:

The assignment operator (=) is used to assign a value to a variable whereas the equal to operator (==) is used to compare two values of same data type.

For example:

A=5;

C=b;

Z=x+y;

In the above statements, variable a is assigned the value 5, c is assigned the value stored in variable b and z is assigned the sum of values stored in variable x and y.

The relational operator (= =) is used to build a condition based on which computer takes some action.

For example:

A= = 1

C= = a+ b

In the first condition, if the value of a is equal to 1 then the condition is true otherwise it is false. In the second condition, the equal to operator is used to check whether the value of c is equal to the sum of a and b. If it is equal then the condition is true otherwise, it is false.

Q13. Differentiate between Unary and Binary Operators.

Ans: Difference between Unary and Binary Operators:

The operators that work with a single operand are known as unary operators whereas operators that work with two operands are known as binary operators. Unary operators are -, ++, -- and the logical operator !(NOT). Binary operators are -, +, *, \, % and logical operators && (AND) and II (OR).

Some examples of unary operators are:

A= -b;

K++;

--x;

Some examples of binary operators are

A= b + c;

Z= x*y;

K=d%e;

Q14. Define and explain the order of precedence of operators.

Ans: Order of Precedence of Operators

Order of precedence of operators is the rule that specifies the order in which operations are to be performed in an expression. The order of precedence is similar to that used in algebraic formulas.

The order of precedence of operators is shown in table. The operator that has the highest precedence is written at the top and the one with the lowest precedence is written at the bottom.

Precedence	**Operator**	**Description**
1.	++, --	Increment and Decrement (Prefix or Postfix)
2.	*, /, %	Multiplication, Division and Remainder
3.	+, -	Addition and Subtraction
4.	<, <=, >, >=	Relational Operators
5.	= =, !=	Equal to and Not Equal to
6.	!	Logical NOT
7.	&&	Logical AND
8.	||	Logical OR
9.	=, *=, /=, +=, -=	Assignment Operators

As shown in table, the increment/decrement operators are performed first, then arithmetic operators, then relational operators, after that logical operators and the end assignment operators.

If an expression contains two or more operators of the same precedence then the operations are performed from left to right in the order in which they appear.

To illustrate this, consider the expression

7* 10 – 5 % 3 * 4 + 9 / 3

The leftmost multiplication is performed first, giving the intermediate result.

70 – 5 % 3 * 4 + 9 / 3

Multiplication is performed next, giving

70 – 8 + 9 / 3

The last high-priority operation , division is performed next, It gives

70 – 8 + 3

Next the low-priority operations are performed in the order in which they occur from left to right. The subtraction Is thus performed first giving.

62 + 3

And then addition is carried out, giving the final result 65

The standard order of evaluation can be modified by using brackets to enclose part of an expression. The part of the expression within brackets is first evaluated in the standard manner and then the result is combined to evaluate the complete expression.

For example in the expression **a/ (b + c)**

B+c will be perfumed first and then a will be divided by the sum of b and c.

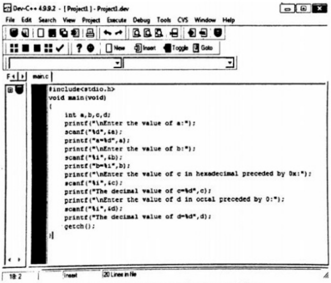
If the brackets are nested, that is, if one set of brackets is contained within another, the computations in the innermost brackets are performed first.

Q15. Why Integer Format Specifier is used? Explain with examples.

Ans: Integer Format Specifier (%d, %ld and %i):

The format specifier %d is used to read or print a decimal integer and the format specifier %id is used with long integers.

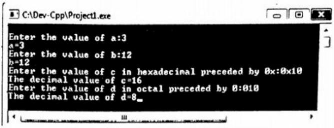
The program in Fig, demonstrates the use of integer format specifiers.



```
#include<stdio.h>
void main(void)
{
    int a,b,c,d;
    printf("\nEnter the value of a:");
    scanf("%d",&a);
    printf("a=%d",a);
    printf("\nEnter the value of b:");
    scanf("%i",&b);
    printf("b=%i",b);
    printf("\nEnter the value of c in hexadecimal preceded by 0x:");
    scanf("%i",&c);
    printf("The decimal value of c=%d",c);
    printf("\nEnter the value of d in octal preceded by 0:");
    scanf("%i",&d);
    printf("The decimal value of d=%d",d);
    getch();
}
```

Program that uses integer format specifiers

The format specifier %d and %i are same when they are used with the printf() function but different when used with scanf() functions. For printf() function, both %d and %i are used for decimal integer as shown in program.



```
Enter the value of a:3
a=3
Enter the value of b:12
b=12
Enter the value of c in hexadecimal preceded by 0x:0x10
The decimal value of c=16
Enter the value of d in octal preceded by 0:010
The decimal value of d=8_
```

The execution of the program is shown in Fig.

Q16. Why Floating-point Format Specifiers is used? Explain with examples.

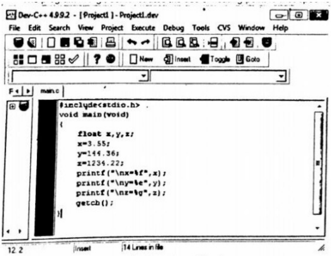
Ans: Floating-point Format Specifiers (%f, %e and %g)

The format specifier %f is used to read and print floating-point numbers in decimal notation with a precision of 6 digits after the decimal point.

The format specifier %e is used to read and print floating-point numbers in exponential notation.

The formal specifier %g is used to print floating-point numbers in decimal or exponential notation whichever is shorter.

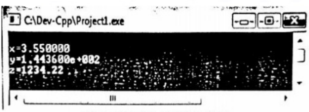
The program in Fig. demonstrates the use of floating-point format specifiers.


Dev-C++ 4.9.9.2 - [Project1] - Project1.dev
File Edit Search View Project Execute Debug Tools CVS Window Help
New Insert Toggle Goto
main.c
12: 2
Insert
14 Lines in file

```
#include<stdio.h>
void main(void)
{
    float x,y,z;
    x=3.55;
    y=144.36;
    z=1234.22;
    printf("\nx=%f",x);
    printf("\ny=%e",y);
    printf("\nz=%g",z);
    getch();
}
```

Usingfloating - pointformatspecifiersinaprogram

The execution of the program is shown in Fig.



```
x=3.550000
y=1.443600e+002
z=1234.22
```

Printingfloating - pointnumbers

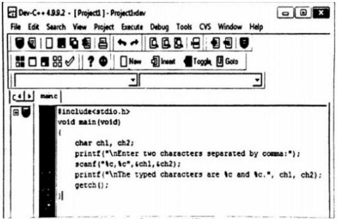
The value of variable z is printed as a floating point number because it is shorter than exponential notation.

Q17. Why Character Formal Specifier is used? Explain with examples.

Ans: The Character Format Specifier (%c):

The character formal specifier, % is used to read or print a single character.

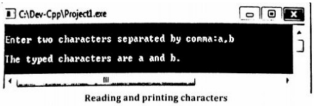
The program in Fig, demonstrates the use of % c format specifier.


Dev-C++ 4.9.9.2 - [Project1] - Project1.dev
File Edit Search View Project Execute Debug Tools CVS Window Help
New Insert Toggle Goto
main.c

```
#include<stdio.h>
void main(void)
{
    char ch1, ch2;
    printf("\nEnter two characters separated by comma:");
    scanf("%c,%c",&ch1,&ch2);
    printf("\nThe typed characters are %c and %c.", ch1, ch2);
    getch();
}
```

Using character format specifier in a program

The execution of the program is shown in Fig.



```
Enter two characters separated by comma:a,b
The typed characters are a and b.
```

Reading and printing characters

KEY POINTS

- The **printf()** functions is used to print text and values on the screen in a specified format.
- The **Putchar()** function is used to output a single character to the screen.
- The **Puts()** function is used to print a string on the screen.
- The **scanf()** function is used to get values into variables from the keyboard during execution of a program
- The **getche()** and getch() functions are used to read a single character from the keyboards, the instant it is typed without waiting for the Enter to be pressed. The getche() function displays the typed character on the screen whereas getch() function does not display it.
- The **getchar()** function is used to read a single character from the keyboard.
- The **gets()** function is used to read a character string from the keyboard.
- A **semicolon** is entered at the end of each C statement to terminate it.
- A **format specifier** tells about the data type, field-width and the format according to which a value is to be printed or read from an input device.
- **Escape sequence** is a combination of backslash (\) and a code character to control printing of data on the screen.
- **Assignment operator** is used to assign a value to a variable.
- **Relational operator** is used to compare two values of the same type. It is used in an expression when a decision is to be based on a condition in a program.
- **Logical operator** is used for building compound condition.

- The **order of precedence** of operators is the rule that specifies the order in which operations are to be performed in an expression.

ClassNotes

EXERCISE

Q1. Select the best answers for the following MCQs.

i. Which function is used for output purpose in C language?

A. printf()
B. scanf()
C. input()
D. getch()

ii. Which character terminates a C statement?

A. Colon
B. Semicolon
C. Period
D. Comma

iii. Which format specifier is used to print or read a floating-point value?

A. %d
B. %i
C. %f
D. %e

iv. Which escape sequence is used to move cursor to the beginning of current line?

A. \a
B. \r
C. \n
D. \b

v. Which of the following is an arithmetic operator?

A. %
B. <=
C. &&
D. +=

vi. Which of the following is a logical operator?

A. %
B. <=
C. &&
D. +=

vii. Which statement is equivalent to "k= k + a;"?

A. k+=a
B. k=+a;
C. k++a;
D. k=a++;

viii. Which of the following is an increment operator?

A. +
B. +=
C. ++

D. =+

ix. **Which of the following operator has the highest precedence?**

A. &&

B. <=

C. =

D. *

x. **What will be the output of the expression, 5+3*3–1?**

A. 16

B. 13

C. 23

D. 12

Answers

i.	**A**	**ii.**	**B**	**iii.**	**C**	**iv.**	**B**	**v.**	**A**
vi.	**C**	**vii.**	**A**	**viii.**	**C**	**ix.**	**D**	**x.**	**B**

SHORT QUESTIONS

Q2. Give short answers to the following questions.

i. Why format specifier is used? Explain with examples.

Ans: Format Specifiers:

A format specifier is a computer code that tells about the data type, field width and the format according to which a value is to be printed or read from an input device. They are used in C language for input and output purposes. Using these format specifiers, compiler can understand the type of data that is in a variable during taking input using the scanf() function and printing output using the printf() function.

A list of commonly used format specifiers is given below

%d	decimal integer
%i	integer
%ld	long decimal integer
%f	floating-point (decimal notation)
%e	floating-point (exponential notation)
%c	single character
%s	string

ii. Why escape sequence is used? Explain with examples.

Ans: Escape sequence:

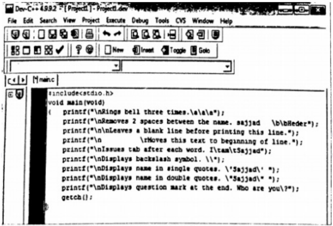
The special characters used in C language to control printing on the output device are called escape sequences. These characters are not printed. These are used inside the control string.

An escape sequence is a combination of a backslash (\) and a code character. The backslash is called the control character. A list of commonly used escape sequence is given in table with their meanings.

Escape sequences in C language:

Escape Sequence	Meaning
\a	Produces alert (bell) sound
\b	Moves cursor backward by one position
\n	Moves cursor to the beginning of next line
\r	Moves cursor to the beginning of current line
\t	Moves cursor to the nest horizontal tabular position
\\	Produces a backslash
\'	Produces a single quote
\ ''	Produces a double quote
\?	Produces a question mark

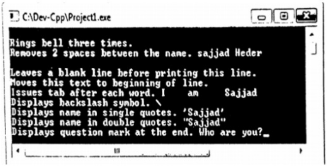
The program in Fig demonstrates the use of escape sequence


Dev-C++ 4.9.9.2 - [Project1] - Project1.dev
File Edit Search View Project Execute Debug Tools CVS Window Help
New
Insert
Toggle
Goto
main.c

```
#include<stdio.h>
void main(void)
{    printf("\nRings bell three times.\a\a\a");
     printf("\nRemoves 2 spaces between the name. sajjad   \b\bHeder");
     printf("\n\nLeaves a blank line before printing this line.");
     printf("\n          \rMoves this text to beginning of line.");
     printf("\nIssues tab after each word. I\tam\tSajjad");
     printf("\nDisplays backslash symbol. \\");
     printf("\nDisplays name in single quotes. \'Sajjad\' ");
     printf("\nDisplays name in double quotes. \"Sajjad\" ");
     printf("\nDisplays question mark at the end. Who are you\?");
     getch();
```

Using escape sequences in a program

The output of the program is shown in Fig.



```
Rings bell three times.
Removes 2 spaces between the name. sajjad Heder

Leaves a blank line before printing this line.
Moves this text to beginning of line.
Issues tab after each word. I    am     Sajjad
Displays backslash symbol. \
Displays name in single quotes. 'Sajjad'
Displays name in double quotes. "Sajjad"
Displays question mark at the end. Who are you?_
```

iii. What is the purpose of printf() function? Explain with an example.

Ans: The printf() Function:

The printf() function is used to print text, constants, values of variables and expressions on the screen in a specified format. Its basic purpose is to output information onto the computer screen. The general syntax of this function is

printf(control_string, list of arguments);

The **control_string** consist of text, format specifiers and escape sequences written within double quotes whereas the **list of arguments** consists of a list of all the variables, constants and arithmetic expressions, separated by commas, whose values are to be printed. Example of printf() function is as follow.

printf("Value of a is %d and b is %d.",a,b);

If the values stored in a and b are 1 and 2 respectfully then the following output will be printed on the screen.

Value of a is 1 and b is 2.

---

iv. Differentiate between printf() and scanf() functions.

Ans: The printf() and scanf() Functions:

printf() function	**scanf() function**
1. The printf() function is used to print text and values on the screen in a specified format.	**1.** The scanf() function is used to get values into the variables from the keyboard during execution of the program.
2. printf() is used to output data.	**2.** scanf() is used to input data.
3. In printf() we pass the variable values as the argument.	**3.** In scanf() we pass the variable addresses as the argument.
4. General syntax of the function is **printf(control_string, list of arguments);**	**4.** General syntax of the function is **Scanf("format specifiers", list of arguments);**

v. Evaluate the following expression.

a) 7+5*(3+4)

b) 100/10/4

c) 50%13%3

d) 30/7*3-6

Ans:

a) 7+5*(3+4)

=7+5*7

=7+35

=42

b) 100/10/4

=10/4

=2.5

c) 50%13%3

=11%3

=2

d) 30/7*3-6

=4.28*3-6

=12.8-6

=6.8

vi. What will be the output of the following program?

```
#include <stdio.h>
void main(void)
{
         int x,y,z1,z2,z3,z4;
         x=17;
         y=5;
         z1=x/y
         printf(“\nz1=%d”,z1);
         z2=x%y;
          printf(“\nz2=%d”,z2);
          z3=++x;
          printf(“\nz3=%d”,z3);
          z4=y++;
         printf(“\nz4=%d”,z4);
}
```

Ans: Output:

```
z1 = 3
z2 = 2
z3 = 18
z4 = 5
```

vii. What will be the output of the following program?

```
#include <stdio.h>
void main(void)
{
         int b;
        float a,c,d,e,f;
        a=14.84;
        b=7;
        c=a-b;
        printf(“\nc=%f”,c);
        d=a/b;
```

```
            printf("\nd=%f",d);
            e=a-b*3
            printf("\ne=%f",e);
            f=(a+b)/2;
            printf("\nf=%f",f);
    }
```

Ans: Output:

```
c= 7.840000
d= 2.120000
e= 6.160000
f= 10.920000
```

EXTENSIVE QUESTIONS

Q3. Describe how basic and compound assignment operators are used?

Ans: Assignment Operators:

Assignment operators are used to assign values to variables used in computer programs. C language provides three types of assignment operators. There are basic assignment operators, compound assignment operators and increment/decrement operators.

Basic Assignment Operator:

The basic assignment operator is =. This is used to assign value of an expression to a variable. It has the general form:

Variable=expression

Where expression may be a constant, another variable to which a value has previously been assigned or a formula to be evaluated. For example:

Sum= a + b;

Compound Assignment Operators:

In addition to =, there are number of assignment operators unique to C. These include +=, -=, /= and %=. Suppose op represents an arithmetic operator. Then, the compound assignment operator has the following general form to assign value of an expression to a variable.

Variable op= expression

For example, consider the following statement:

Sum= sum + n;

This assignment statement could be written using a compound assignment operator as:

Sum += n;

The effect is exactly the same but the expression is more compact.

Some more examples are:

Sum-= n	**is equivalent to**	**sum= sum – n**
Prod * = n	**is equivalent to**	**prod = prod *n**
A / = b	**is equivalent to**	**a = a / b**
A % = b	**is equivalent to**	**a = a % b**

Q4. Describe the functions of the following operators?

i) Relational operators
ii) Logical operators
iii) Conditional operator

Ans: i) Relational operators:

Relational operators are used to compare two values of the same type. These are used in expressions when a decision is to be based on a condition. After evaluation of a relational expression, the result produced is either True or False. Relational operators are used in programming for decision making.

Types of Relational Operators:

Six types of relational operators are available in C language. These are described in table

Operator	**Definition**
==	Equal to
!=	Not equal to
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

The following are some examples of relational operators

c>=a+b

x<5.3

n==20

count !=10

If a has the value 25, b has the value 10 and c has the value 28 then first expression is false since 28 is not greater than or equal to 35.

In the second expression, if x has the value 4.5 the expression is true because x is less than 5.3

In the third expression, if n is equal to 20 then the expression will be true. For any value other than 20, the expression will be false.

In the last expression if count is any number other than 10 then the expression will be true. It will only be false when count is equal to 10.

ii) Logical operators:

Logical operators are used for building compound conditions. We have seen before that a single condition is built using a relational operator in an expression. If we need to build more than one condition for some action to take place in programming, then we have to form compound condition.

Types of Logical Operators:

There are three types of logical operators. These are described in table.

Operator	Definition
&&	AND
II	OR
!	NOT

Logical AND (&&) Operator:

It is used to form compound condition in which two relational expressions are evaluated. One relational expression is to the left and the other to the right of the operator. If both of the relational expressions (conditions) are true then the compound condition is considered true otherwise it is false.

Syntax:

Expression 1 && Expression 2

Truth table for AND operator is shown here under:

Expression-1	Expression-2	Expression1 && Expression-2
True	True	True
True	False	False
False	True	False
False	False	False

Example:

Consider the following compound condition:

(a>=1) && (a<=10)

This expression is considered true of both conditions (a>=1) and (a<=10) are true, that is, if the value of a is between 1 to 10.

In the next example, the compound condition is considered true if the value stored in a is greater than the value stored in b and the value stored in c is equal to 15.

(a>b) && (c= =15)

The following compound condition will check whether the character stored in variable **ch** is a lowercase letter or not.

(ch>='a') && (ch<='z')

Logical OR (I I) Operator:

Logical OR operator is also used to form a compound condition.

Just like the logical AND operator, one relational expression is to the left and the other to the right of the OR operator. The compound condition is true if either of the conditions is true or both conditions are true. It is considered false only if both of the countries are false.

Syntax:

Expression 1 I I Expression 2

Truth table for AND operator is shown here under

Expression-1	**Expression-2**	**Expression-1 I I Expression-2**
True	True	True
True	False	True
False	True	True
False	False	False

Example:

(n<10)II(n>25)

Suppose the value of n is 5, then the expression will be considered true because one of the two conditions is true. If the value of n is 28 then also the compound condition will be true. If the value of n is 12 then the expression will be false since both conditions are false.

The next compound condition will be true if a is greater than b or c is equal to 10. It will also be true if both conditions are true, that is, a is greater than b and c is equal to 10. It will only be false if a is not greater than b and at the same time c is not equal to 10.

(a<b) II (C=10)

Logical OR condition is used when we wish to perform an operation if one of two conditions is true or both of the countries are true.

Logical NOT (!) Operator

The logical NOT operator is used with a single expression (condition) and evaluates to true if the expression is false and evaluates to false if the expression is true. In other words, it reverses the result of a single expression.

Syntax:

!Expression1

Truth table for AND operator is shown here under:

Expression	!Expression
True	False
False	True

For example, the expression:

!(a>b)

Will be true if a is not less than b. In other words, the condition will be true if a is greater than or equal to b. The same condition can also be written as given below which is easy to understand

(a>=b)

i. Conditional operator/ Conditional (Ternary) Operator:

A conditional operator is a decision-making operator. It has the following form.

Condition?expression1 : expression2:

When this statement is executed the condition is evaluated. If it is true, the entrie conditional expression takes on the value of expression1. If it is false, the conditional expression takes on the value of expression2. The entire conditional expression takes on a value and can therefore be used in an assignment statement. Consider the following example.

A= (k>15)? X*y : x+y;

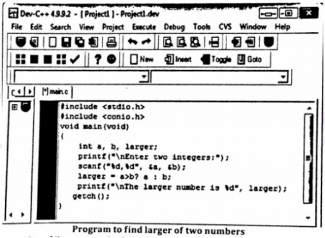
This statement will assign the product of x and y to the variable a, if k is greater than 15, otherwise a will be assigned the sum of x and y. This expression is equivalent to the following if-else statement which will be explained in the next unit.

```
If (k>15)
```

```
        A= x*y;
Else
        A= x+y;
```

Some programmers may prefer to use the above if else statement rather than using the conditional operator because it is easy to understand.

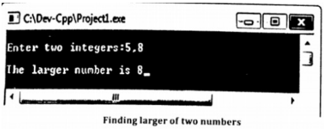
The program in Fig, demonstrate the use of conditional operator for finding the larger of two numbers.


Dev-C++ 4.9.9.2 - [Project1] - Project1.dev
File Edit Search View Project Execute Debug Tools CVS Window Help
New Insert Toggle Goto
[*] main.c

```
#include <stdio.h>
#include <conio.h>
void main(void)
{
    int a, b, larger;
    printf("\nEnter two integers:");
    scanf("%d,%d", &a, &b);
    larger = a>b? a : b;
    printf("\nThe larger number is %d", larger);
   getch();
}
```

Program to find larger of two numbers

The execution of the program is shown in Fig.



```
Enter two integers:5,8
The larger number is 8_
```

Finding larger of two numbers

Q5. Write a program that reads three numbers and prints their sum, product and average.

Ans: Program for Sum, Product and Average

```
#include <stdio.n>
void main(void)
{
  int num1, num2, num3, sum, prod, avg;
  printf(“Enter the three numbers separated by ‘,’\n”);
  scanf(“%d, %d, %d”, &num1, &num2, &num3);
  sum= num1+ num2 + num3;
  prod=num1 * num2 * num3;
  avg=sum/3
  printf(“the sum of the three numbers is : %d\n”, sum);
  printf(“the prod of the three numbers is : %d\n”, prod);
  printf(“the average of the three numbers is : %d\m”, avg);
```

```
}
```

Q6. Write a program that reads the length and width of a rectangle and prints its area.

Ans: Program for Area of a Rectangle

```
#include <stdio.h>
void main(void)
{
  float length, width, area;
  printf(“Enter length of rectangle: ");
  scanf(“%f”,&length);
  printf(“Enter width of rectangle:");
  scanf(“%f”,&width);
  area= length * width;
```

```
  printf(“Area of rectangle = %f sq.units” , area);
}
```

Q7. Write a program that reads the length of one side of a cube and prints its volume.

Ans: Program for Volume of a Cube

```
#include <stdio.h>
#include <conio.h>
void main(void)
```

```
{
    float side, volume;
    printf(“Enter length of any side of cube\n”);
    scanf(“%f, &side);
    volume = side*side*side;
    printf (“Volume of Cube : %0.4f\n”, volume);
}
```

Q8. Write a program that reads temperature in Celsius, converts it into Fahrenheit and prints on the screen.

Ans: Program to Convert Temperature from Celsius to Fahrenheit

```
#include <stdio.h>
#include <conio.h>
void main(void)
{
   float farhren, Celsius;
   printf(“Enter the temperature in Celsius\n”);
   scanf(“%f”, &Celsius);
   fahren = (9.0/5.0) * Celsius + 32;
   printf(“%.2fc is equal to %.2F\n”, Celsius, fahren);
}
```