Q.2 a. Classify computers in different categories depending on the computing ability and processing speed. **(6)**

Answer

Computers can be classified into several categories depending on their computing ability and processing speed. These include:

- **Microcomputers:** A microcomputer is defined as a computer that has a microprocessor as its CPU. The microcomputer system can perform the following basic operations:
  - **Inputting:** It is the process of entering data and instructions into microcomputer system.
  - **Storing:** It is the process of saving data and instructions in the memory of the microcomputer system, so that they can be use whenever required.
  - **Processing:** It is the process of performing arithmetic or logical operations on data, where data can be converted into useful information. Various arithmetic operations include addition, subtraction, multiplication and division. Among logical operations, operations of comparisons like equal to, les than etc., are prominent in use.
  - **Outputting:** It provides the results to the user, which could be in the form of visual display and/or printed reports.
  - **Controlling:** It helps in directing the sequence and manner in which all the above operations are performed.

- **Minicomputers:** A minicomputer is a medium-sized computer that is more powerful than a microcomputer. An important distinction between a microcomputer and a minicomputer is that a minicomputer is usually designed to serve multiple users simultaneously. A system that supports multiple users is called a multiterminal, time-sharing system. Minicomputers are the popular computing systems among research and business organizations today. They are more expensive than microcomputers.
- **Mainframe Computers:** Mainframe computers are those computers, which help in handling the information processing of various organizations like banks, insurance companies, hospitals and railways. Mainframe computers are placed on a central location and are connected to several user terminals, which can act as access stations and may be located in the same building. Mainframe computers are larger and expensive in comparison to the workstations.
- **Supercomputers:** Supercomputers are the most powerful and expensive computers available at present. They are also the fastest computers available. Supercomputers are primarily used for complex scientific applications, which need a higher level of processing. Some of these applications include weather forecasting, climate research, molecular modelling used for chemical compounds, aeroplane simulations and nuclear fusion research.

In supercomputers, multiprocessing and parallel processing technologies are used to promptly solve complex problems. Here, the multiprocessor can enable the user to divide a complex problem into smaller problems. A supercomputer also supports multiprogramming where multiple users can access the computer simultaneously.

b. Differentiate between System software and Application software. (5)

Answer

System Software

System software refers to a computer program that manages and controls hardware components of a computer system. In other words, the system software is responsible for handling the functioning of the computer hardware. The system software is also responsible for the proper functioning of the application software on a computer system. The system software includes general programs, which are written to provide an environment for developing new application software using programming languages. In computer science, there are several types of system software, such as operating systems and utility programs. The operating system is the primary system software, which controls the hardware and software resources of a computer system. It also performs various operations, such as memory allocation, instruction processing, and file management. The most commonly used operating systems are MS DOS, UNIX, etc. The following are the various functions of system software:

- Process management
- Memory management
- Secondary storage management
- I/O system management
- File management

Application Software

Application software is a computer program that is executed on the system software. It is designed and developed for performing tasks and is also known as end-user program. Application software is unable to run without system software, such as operating system and utility programs. It includes several applications, such as word-processing and spreadsheet. The word-processing application helps in creating and editing document. Using application software, we can also format and print the document.

c. What is scanner? What are the different types of scanners that can be used to produce digitized images? (5)

Answer

A scanner is an input device that converts documents and images as the digitized images understandable by the computer system. The digitized images can be produced as black and white images, gray images, or colored images. In case of colored images, an image is considered as a collection of dots with each dot representing a combination of red, green and blue colors, varying in proportions. The proportions of red, green, and blue colors assigned to a dot are together called as color description. The scanner uses the color description of the dots to produce a digitized image.

Following are the types of scanners that can be used to produce digitized images:

- **Flatbed scanner:** It contains a scanner head that moves across a page from top to bottom to read the page and converts the image and text available on the page in digital form. The flatbed scanner is used to scan graphics, oversized documents, and pages from books.

- **Drum scanner:** In this type of scanner, a fixed scanner head is used and the image to be scanned is moved across the head. The drum scanners are used for scanning prepress materials.
- **Slide scanner:** It is a scanner that can scan photographic slides directly to produce files understandable by the computer.
- **Handheld scanner:** It is a scanner that is moved by the end user across the page to be scanned. This type of scanner is inexpensive and small in size.

Q.3 a. Write the BCD equivalent of decimal digits 0 – 9. (4)

Answer

In BCD code, each decimal digit is represented by a binary code of four bits, and the binary weights of four bits are 2^3, 2^2, 2^1 and 2^0. The decimal digits and corresponding BCD numbers are shown in following table.

Decimal Digit	BCD Equivalent
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

b. What is an operating system? What are the various categories of operating systems? (6)

Answer

An operating system can be defined as the system software that helps in managing the resources of a computer as well as provides a platform for the application programs running in the computer. In other words, the operating system acts as an interface between the computer and its applications programs.

Depending on the characteristics of operating systems, they can be categorized into the following types:

- **Batch operating system:** This is the earliest operating system, where only one program is allowed to run at one time. We cannot modify any data used by the program while it is being run. If an error is encountered, it means starting the program from scratch all over again. A popular batch operating system is MS DOS.
- **Interactive operating system:** This operating system comes after the batch operating system, where also only one program can run at one time. However, here, modification and entry of data are allowed while the program is running. An example of an interactive operating system in Multics (Multiplexed Information and Computing Service).

- **Multiuser operating system:** A multiuser operating system allows more than one user to use a computer system either at the same time or at different times. Examples of multiuser operating systems include Linux and Windows 2000.
- **Multi-tasking operating system:** A multi-tasking operating system allows more than one program to run at the same time. Examples of multi-tasking operating system include UNIX and Windows 2000.
- **Multithreaded operating system:** A multithreaded operating system allows the running of different parts of a program at the same time. Examples of multithreaded operating system include UNIX and Linux.

c. Differentiate memory units on the basis of Primary memory and Secondary memory. (6)

Answer

The primary memory

The primary memory is available in the computer as a built-in unit of the computer. The primary memory is represented as a set of locations with each location occupying 8 bits. Each bit in the memory is identified by a unique address. The data is stored in the machine-understandable binary form in these memory locations. The commonly used primary memories are as follows:

- **ROM:** ROM represents Read Only Memory that stores data and instructions, even when the computer is turned off. It is the permanent memory of the computer where the contents cannot be modified by an end user. ROM is a chip that is inserted into the motherboard. It is generally used to store the Basic Input/Output system (BIOS), which performs the Power On Self Test (POST).
- **RAM:** RAM is the read/write memory unit in which the information is retained only as long as there is a regular power supply. When the power supply is interrupted or switched off, the information stored in the RAM is lost. RAM is a volatile memory that temporarily stores data and applications as long as they are in use. When the use of data or the application is over, the content in RAM is erased.
- **Cache Memory:** Cache memory is used to store the data and the related application that was last processed by the CPU. When the processor performs processing, it first searches the cache memory and then the RAM, for an instruction. The cache memory can be either soldered into the motherboard or is available as a part of RAM.

The secondary memory

Secondary memory represents the external storage devices that are connected to the computer. They provide a non-volatile memory source used to store information that is not in use currently. A storage device is either located in the CPU casing of the computer or is connected externally to the computer. The secondary storage devices can be classified as:

- **Magnetic storage device:** The magnetic storage devices store information that can be read, erased and rewritten a number of times. These include floppy disk, hard disk and magnetic tapes.

- **Optical storage device:** The optical storage devices are secondary storage devices that use laser beams to read the stored data. These include CD-ROM, rewritable compact disk (CD-RW), digital video disks with read only memory (DVD-ROM), etc.
- **Magneto-optical storage device:** The magneto-optical devices are generally used to store information, such as large programs, files, and back up data. The end user can modify the information stored in magneto-optical storage devices multiple times. These devices provide higher storage capacity as they use laser beams and magnets for reading and writing data to the device.

Q.4 a. Define algorithm. Write an algorithm to find the factors of a given number. (2+4)

Answer

An algorithm is a complete, detailed, and precise step-by-step method for solving a problem independently of the software or hardware of the computer. Algorithms are very essential, as they instruct the computer what specific steps it needs to perform to carry out a particular task or to solve a problem.

<u>Algorithm to find the factors of a given number</u>

```
1. Read a number, say num.
2. If num < 0, then goto step 11.
3. Set i=1.
4. Repeat step 5 to 10.
5. If i > num, then goto 10.
6. Else
7. Divide num by i.
8. If the remainder of the division is 0, print i.
9. Increment i by 1 and goto step 5.
10. Endif.
11. Exit
```

b.Describe the structure of a C program. (5)

Answer

A C program can be viewed as a group of building blocks called functions. A function is a subroutine that may include one or more statement designed to perform a specific task. To write a C program, we first create functions and then put them together. A c program may contain one or more sections.

The documentation section consists of a set of comment lines giving the name of the program, the author and other details, which the programmer would like to use later. The link section provides instructions to the compiler to link functions from the system library. The definition section defines all symbolic constants.

There are some variables that are used in more than one function. Such variables are called global variables and are declared in the global declaration section that is outside of all the functions. This section also declares all the user-defined functions.

Every C program must have one **main()** function section. This section contains two parts, declaration part and executable part. The declaration part declares all the variables used in the executable part. There is at least one statement in the executable part. These two

parts must appear between the opening and the closing braces. The program execution begins at the opening brace and ends at the closing brace. The closing brace of the main function section is the logical end of the program. All statements in the declaration and executable parts end with a semicolon(;).
The subprogram section contains all the user-defined functions that are called in the main function. User-defined functions are generally placed immediately after the main function, although they may appear in any order.
All sections, except the main function section may be absent when they are not required

c. Explain the process of compiling and linking multiple source program files under UNIX environment. (5)

Answer

To compile and link multiple source program files, we must append all files names to the cc command as

```
cc filename-1.c, ......, filename-n.c
```

These files will be separately compiled into object files called

```
file-name-i.o
```

and then linked to produced an executable file say, a.out.

It is also possible to compile each file separately and link them later. For examples, the commands

```
cc –c mod1.c
cc –c mod2.c
```

will compile the source files mod1.c, mod2.c into objects files mod1.o and mod2.o. They can be linked together by the command

```
cc mod1.o mod2.o
```

We can also combine the source files and object files as

```
cc mod1.c mod2.o
```

Only mod1.c is compiled and linked with the object file mod2.o. This approach is useful when one of the multiple source files need to be changed and recompiled or an already existing object files is to be used along with the program to be compiled.

Q.5 a. Write various data types supported in C programming language with examples. (4)

Answer

Data types supported in C are as follows:

Name of data type	**Keyword**	**Description**	**Example**
Character	char	Declares character type variables	char ch;
Integer	int, short int,	Declares integer type	int a = 5;

	long int	variables	short int x;
			long int y;

Floating point	float, double, long double	Declares real type variables	float z=5.5;
			double salary;
			long double q;

Void	void	Used to specify void or no values	void funt1(int x) {
			....
			Body of the function
			....
			}

b. What is variable? What are the rules must be followed while naming a variable name? (4)

Answer

A variable is a data name that may be used to store a data value. Unlike constants that remain unchanged during execution of a program, a variable may take different values at different times during execution. A variable name can be chosen by the programmer in a meaningful way so as reflect its function or nature in the program. Some examples of variable names are as follows:

Height, salary, counter, average etc.

Variables names may consist of letters, digits, and the underscore (_) character, subject to the following conditions:

- They must begin with a letter. Some systems permit underscore as the first character.
- ANSI standard recognizes a length of 31 characters.
- Uppercase and lowercase are significant. The variable Total and total are different.
- It should not be a keyword.
- White space is not allowed.

c. What is an unsigned integer constant? What is the significance of declaring a constant as unsigned? (4)

Answer

An integer constant is any number in the range -32768 to +32767, because an integer constant always occupies two bytes in memory and in two bytes we cannot store a number bigger than +32767 or smaller than -32768. Out of the two bytes to store an integer, the highest bit is used to store the sign of the integer. This bit is 1 if the number is negative and 0 if number is positive. Unsigned integer constant is an integer constant which has the permissible range from 0 to 65536. Thus significance of declaring a constant as unsigned almost doubles the size of the largest possible value. This happens because on declaring a constant as unsigned, the sixteenth bit is free and is not used to store the sign of the constant.

d. Given the real number y = 98.7654, write the printf statement to get the output as: (4)

(i) 98.77
(ii) 0098.77
(iv) 9.88e+001
(v) 9.876540e+001

Answer

i) printf(“%7.2f”, y);
ii) printf(“%07.2f”, y);
iii) printf(“%10.2e”, y);
iv) printf(“%e”, y);

Q.6 a. Given the values of the variable x, y and z, write a program to rotate their values such that x has the value of y, y has the value of z, and z has the value of x. (5)

Answer

```
#include <stdio.h>

main() {
        int x, y, z, temp;

        printf("\nEnter the value of x, y, z ");
        scanf("%d %d %d", &x, &y, &z);

        temp = x;
        x = y;
        y = z;
        z = temp;

        printf("\nValue of x after rotation is : %d", x);
        printf("\nValue of y after rotation is : %d", y);
        printf("\nValue of z after rotation is : %d", z);
}
```

b. What is the use of bitwise operators? Describe any two bitwise operators with example. (5)

Answer

Bitwise operators are mainly used for manipulation of data at bit level. In addition to it, bitwise operators can also be used for testing the bits and for shifting them for right to left or vice versa. The most common types of bitwise operators are bitwise AND (&), bitwise OR(||), bitwise exclusive OR(^), shift left (<<) and shift right (>>). The following points explain the two bitwise shift operators:

- Shift Left: The left shift operator shifts the operand bits one place to the left and the rightmost bit becomes 0. Example
  Shiftleft(1101) → (1010)

- Shift Right: The right shift operator shifts the operand bits one place to the right and the leftmost bit becomes 0. Example
  Shiftright(1010) → (0101)

c. Explain the following with help of example: (6)
(i) Comma operator
(ii) sizeof operator

Answer

i) Comma operator

The comma operator can be used to link the related expressions together. A comma-linked list of expressions is evaluated left to right and the value of the right-most expression is the value of the combined expression. For example, the statement

```
value = ( x = 10, y = 5, x+y);
```

first assigns the value 10 to x, then assigns 5 to y, and finally assigns 15 to value. Since comma operator has the lowest precedence of all operators, the parentheses are necessary.

ii) sizeof operator

The sizeof operator is a compile time operator and when used with an operand, it returns the number of bytes the operand occupies. The operand may be a variable, a constant or a data type qualifier. Examples

```
m = sizeof(sum);
n = sizeof(float);
p = sizeof(235L);
```

The sizeof operator is normally used to determine the lengths of arrays and structures when their sizes are not known to the programmer. It is also used to allocated memory space dynamically to variables during execution of a program.

Q.7 a. What is function declaration? What are the places in a program where a function declaration can be declared? Is prototype declaration is essential? Give reason. (2+2+2)

Answer

Like variables, all functions in a C program must be declared, before they are invoked. A function declaration, also known as function prototype, consists of four parts as:

- function type (return type)
- function name
- parameter list
- terminating semicolon

The general syntax for function declaration is as follows:

```
function_type function_name(parameter_list);
```

This is very similar to the function header line except the terminating semicolon. For example,

```
int mul(int m, int n);          /* FUNCTION PROTOTYPE */
```

Following points are to be noted:

- The parameter list must be separated by commas.
- The parameter names do not need to be the same in the prototype declaration and the function definition.
- The types must match the types of parameters in the function definition, in number and order.
- Use of parameter names in the declaration is optional.
- If the function has no formal parameters, the parameter list is written as (void);
- The return type is optional, when the function returns int type data.
- The return type must be void if no value is returned.
- When the declared types do not match with the types in the function definition, compiler will produce an error.

Equally acceptable forms of declaration of mul function are:

- int mul(int , int);
- mul(int m, int n);
- mul(int , int);

A prototype declaration may be placed in two places in a program:

1. Above all functions (including main);
2. Inside a function definition.

When we place the declaration above all functions (in the global declaration), the prototype is referred to as a global prototype. Such declarations are available for all the functions in the program.
When we place it in a function definition (in the local declaration), the prototype is called a local prototype. Such declarations are primarily used by the functions containing them.

The place of declaration of a function defines a region in a program in which the function may be used by other functions. This region is known as the scope of the function. It is good programming style to declare prototypes in the global declaration section before main. It adds flexibility, provides an excellent quick references to the functions used in the program, and enhances documentation.

Prototype declarations are not essential. If a function has not been declared before it is used, C will assume that its details available at the time of the linking. Since the prototype is not available, C will assume that the return type is an integer and that types of parameters match the formal definitions. If these assumptions are wrong, the linker will fail and we will have to modify the program. The moral is that we must always include prototype declarations, preferably in global declaration section.

b.What is recursion? Write a C program to calculate X^Y using recursion where values of X and Y are entered through keyboard. Don't use pow() function. **(6)**

Answer

```
#include<stdio.h>
#include<conio.h>

int rec(int,int);

void main() {
        int x,y,res;
        clrscr();
        printf("This program will calculate X^Y ");

        printf("\n\nEnter the value of X: ");
        scanf("%d",&x);

        printf("\nEnter the value of Y: ");
        scanf("%d",&y);

        res=rec(x,y);
        printf("\nThe result is %d",res);
        getch();
}

int rec(int a,int b) {
        int f;
        f=a*rec(a);
}
```

c. **Write a function prime that returns 1 if its argument is a prime number and returns zero otherwise.** **(4)**

```
#include <stdio.h>

int prime(int num);

main() {
        int number;
        int result;
        printf("Enter any number (greater than 2): ");
        scanf("%d", &number);

        result = prime(number);

        if (result == 1)
                printf("\nEntered number %d is a prime number", number);
        else
                printf("\nEntered number %d is not a prime number", number);
}
```

```
int prime (int num) {
        int i,k;
        int flag;
        flag = 0;
        k = (num / 2);

        for(i=2; i<=k+1 ; i++) {
                if (num % i == 0) {
                        flag = 1;
                        break;
                }
        }

        if (flag == 0)
                return 1;
        else
                return 0;
}
```

Q.8 a. Given a number, write a program using while loop to find the sum and reverse the digits of the number where the number is user input. For example, if the number entered is "1234" the sum would be (1+2+3+4) and reverse of the number should be written as 4321. **(6)**

Answer

```
#include <stdio.h>

main() {

        int num, rev, sum, dig;

        printf("\nEnter any number : ");
        scanf("%d",&num);

        rev=0;
        sum=0;

        while (num != 0) {
                dig = num % 10;
                sum = sum + dig;
                rev = rev * 10 + dig;
                num = num / 10;
        }

        printf("Sum of digits is : %d", sum);
        printf("\nReverse of a Number is : %d", rev);
}
```

b. **Explain the following with examples:**
   (i) while statement
   (ii) if ..... else ladder statement
   (iii)continue statement
   (iv) The ? : operator **(2½ ×4)**

Answer

i) while statement

The simplest of all the looping structures is the **while** structure. the basic syntax of the while statement is

```
while (test condition) {
        body of loop
}
```

The while is an entry-controlled loop statement. The test-condition is evaluated and if the condition is true, then the body of the loop is executed. After execution of the body, the test-condition is once again evaluated and if it is ture, the body is executed once again. This process of repeated execution of the body continues until the test-condition finally becomes false and the control is transferred out of the loop. On exit, the program continues with the statement immediately after the body of the loop.

The body of the loop may have one or more statements. The braces are needed only if the body contains two or more statements. However, it is a good practice to use braces even if the body has only one statement.

Suppose we want to calculate the sum of squares of all integers between 1 and 10. We can write the program using while loop as follows:

```
----------------------
----------------------
sum = 0;
n = 1;
while (n <= 10) {
        sum = sum + n * n;
        n++;
}
printf("Sum is :", sum);
-------------------------
-------------------------
```

The body of the loop is executed 10 times for n = 1, 2, 3, ... , 10, each time adding the square of the value of n, which is incremented inside the loop. The test condition may also be written as n<11; the result would be same. The variable n is called counter or control variable. Another example of while statement, which uses the keyboard input, is shown as below:

```
--------------------
--------------------
character = ' ';
while (character != 'Y')
        character = getchar();
-------------------------------
-------------------------------
```

ii) if ..... else ladder statement

if .... else is another way of putting **ifs** together when multipath decisions are involved. A multipath decision is a chain of **ifs** in which the statement associated with each else is an if. It takes the general format as:

```
if (condition 1)
        statement 1;
else if (condition 2)
                statement 2;
else if (condition 3)
                statement 3;
----------------------
-----------------------
else if (condition n)
                statement n;
else
        default statement;

next-statement;
---------------------
```

This construct is known as else if ladder. The condition are evaluated from the top (of the ladder), downwards. As soon as a true condition is found, the statement associated with it is executed and the control is transferred to the next-statement (skipping the rest of the ladder). When all the n conditions become false, then the final else containing the default statement will be executed.

Let us consider an example of grading the students in an academic institution. The grading is done according to the following rules:

Average marks	**Grade**
80 to 100	Honours
60 to 79	First Division
50 to 59	Second Division
40 to 49	Third Division
0 to 39	Fail

This grading can be done using the else if ladder as follows:

```
---------------
---------------
if (marks > 79)
           grade = "Honours";
        else if (marks > 59)
                   grade = "First Division");
                else if (marks > 49)
                           grade = "Second Division";
                        else if (marks > 39)
                                   grade = "Third Division";
                                else
                                    grade = "Fail";
---------------
```

----

iii) continue statement

During the loop operations, it may be necessary to skip a part of the body of loop under certain conditions. For example, in processing of applications for some job, we might like to exclude the processing of data of applicants belonging to certain category.

C supports continue statement to causes the loop to be continued with the next iteration after skipping a part of statements in between. The continue statement tells the compiler "Skip the Following Statements and CONTINUE with the next iteration". The format of the continue statement is simply

```
continue;
```

The use of the continue statement in loops is illustrated as follows:

```
(a)    while (test-condition) {
           ....................
           if (condition)
               continue;
           .............
           .............
       }

(b)    do {
           .....................
           if (condition)
               continue;
           ...................
           ...................
       } while (test-condition);

(c)    for (initialization; test-condition; update) {
           .....................
           if (condition)
               continue;
           ..................
           ..................
       }
```

In all the above three situations if the condition in if-statement is found true continue statement executes and statements following continue statement will be bypassed and the loop will go in the next iteration.

iv) The ? : operator

The C language has an unusual operator, useful for making two-way decision. This operator is a combination of ? and :, and takes three operands. This operator is popularly known as the conditional operator. The general form of use of the conditional is as follows:

(conditional expression) ? expression1 : expression2

The conditional expression is evaluated first. If the result is nonzero, expression1 is evaluated and is returned as the value of the conditional expression. Otherwise expression2 is evaluated and its value is returned.

For example, the segment

```
        if (x < 0)
                flag = 0;
        else
                flag = 1;
```

Q.9 a. Using multidimensional array, write a program in C to sort a list of names in alphabetical order. **(8)**

Answer

A C program to sort a list of names in alphabetical order:

```
#include<stdio.h>
#include<conio.h>
void main() {
        int i,j,n;
        char a[10][20],b[20];
        clrscr();
        printf("how namy names u want to enter");
        scanf("%d",&n);

        printf("enter names :\n");
        for(i=0;i<n;i++)
                scanf("%s",a[i]);

        for(i=0;i<=n-2;i++) {
                for(j=0;j<n-i-1;j++) {
                        if(cmp(a[j],a[j+1])>0) {
                                cpy(b,a[j]);
                                cpy(a[j],a[j+1]);
                                cpy(a[j+1],b);
                        }
                }
        }

        for(i=0;i<n;i++)
                printf("\n%s\n",a[i]);
                getch();
}
```

```
void cpy(char *b,char *a) {
        int i=0;

        while(a[i]!='\0') {
                b[i]=a[i];
                i++;
        }
        b[i]='\0';
}

int cmp(char *a,char *b) {
        int i=0;

        while(a[i]!='\0'||b[i]!='\0') {
                if(a[i]>b[i])
                        break;

                if(b[i]>a[i])
                        break;
                else {
                        i++;
                }
        }
        if(a[i]>b[i])
                return(1);
        if(b[i]>a[i])
                return(-1);
}
```

b. Write a function that accepts two strings str1 and str2 as arguments and finds which of the two is alphabetically greater (without using the library functions). The function should return 1 if str1 is greater than str2, 0 if str1 is equal to str2, and -1 if str1 is smaller than str2.

Answer

A function that accept two strings to check which one is alphabetically greater:

```
void main() {
        char a[10],b[10];
        int k;
        clrscr();

        printf("enter 1st string");
        gets(a);
        printf("enter 2nd string");
        gets(b);

        k=comp(a,b);
        if(k == 1)
                printf("%s is greater than %s",a,b);

        if(k == -1)
                printf("%s is less than %s",a,b);

        if(k == 0)
                printf("%s is equal to %s",a,b);
        getch();
}

int comp(char *a,char *b) {
        int i=0,j=0,k;

        while(a[i] !=’ \0 ' &&  b[j] != '\0’) {
                if (a[i]<b[j])
                        return (-1);
                else if(a[i]>b[j])
                        return (1);
                else
                        i++;
                        j++;
        }

        if(i==j)
                return (0);
        if (i<j)
                return (-1);
        if(i>j)
                return (1);
}
```

TEXT BOOK

Computer Concepts and Programming in C, E. Balagurusamy, Tata McGraw- Hill, 2010