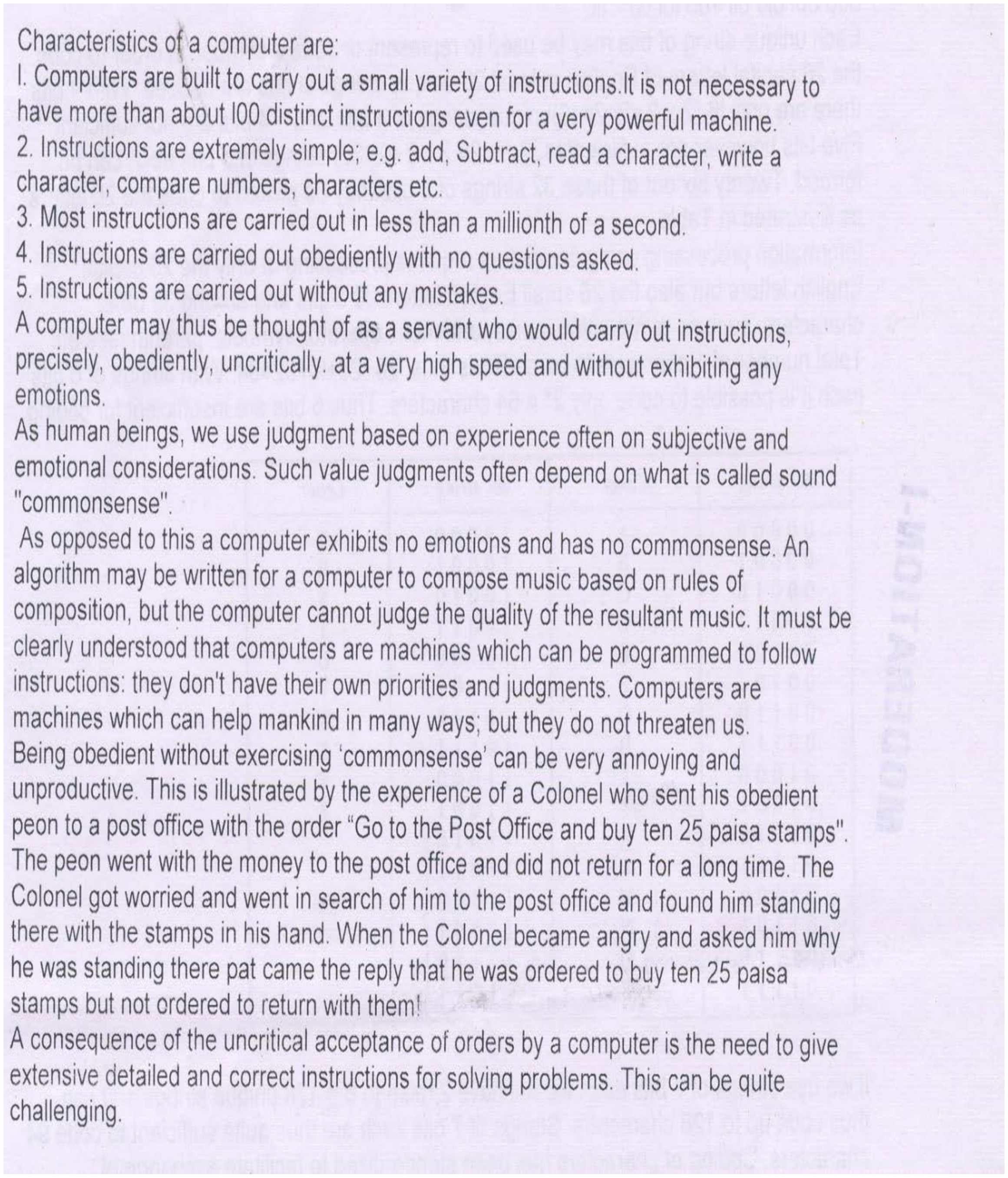
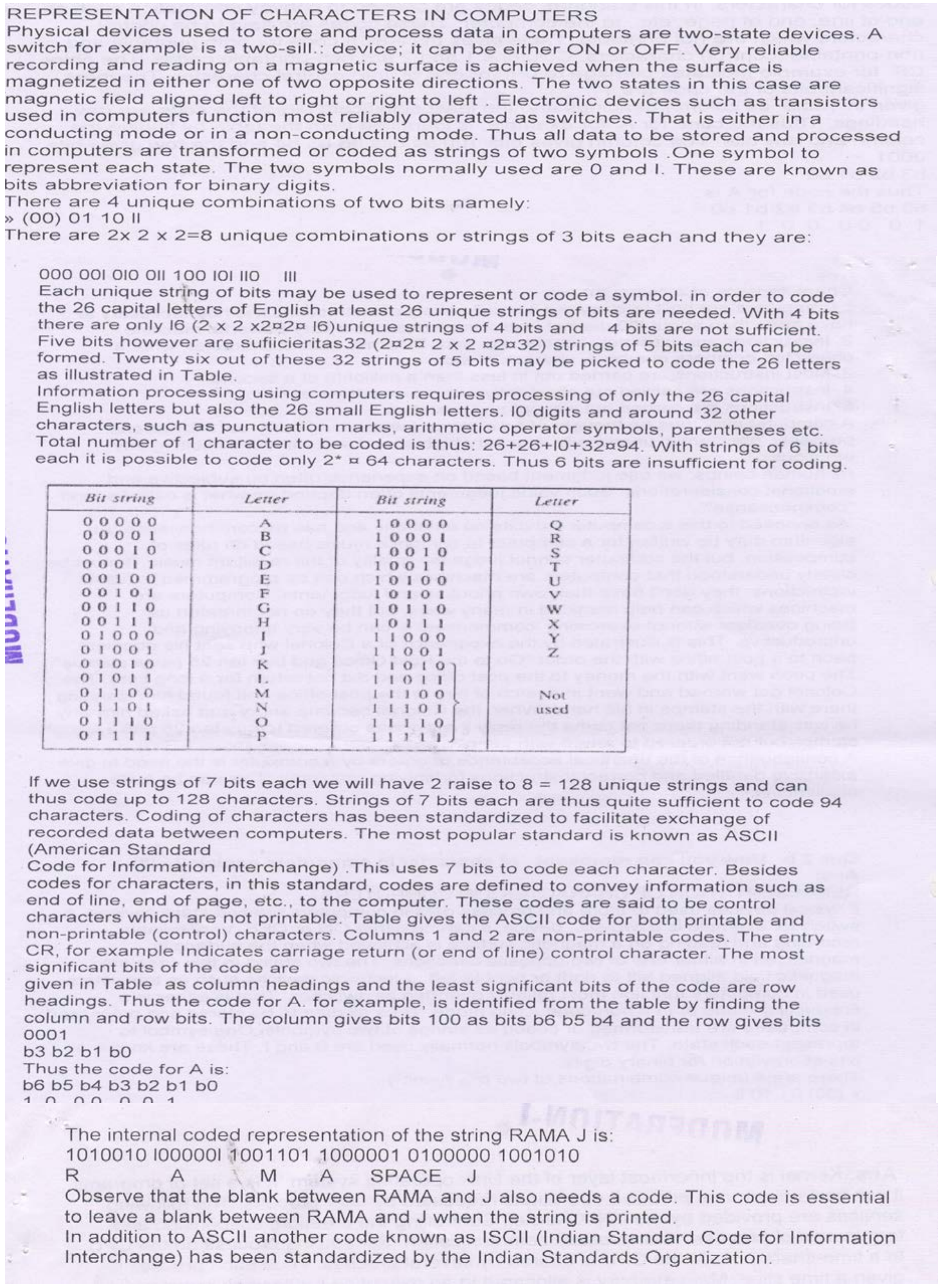
Q2 (a) Discuss the characteristics of computers in brief.

Answer


Characteristics of a computer are:
I. Computers are built to carry out a small variety of instructions.It is not necessary to have more than about I00 distinct instructions even for a very powerful machine.
2. Instructions are extremely simple; e.g. add, Subtract, read a character, write a character, compare numbers, characters etc.
3. Most instructions are carried out in less than a millionth of a second.
4. Instructions are carried out obediently with no questions asked.
5. Instructions are carried out without any mistakes.
A computer may thus be thought of as a servant who would carry out instructions, precisely, obediently, uncritically. at a very high speed and without exhibiting any emotions.
As human beings, we use judgment based on experience often on subjective and emotional considerations. Such value judgments often depend on what is called sound "commonsense".
As opposed to this a computer exhibits no emotions and has no commonsense. An algorithm may be written for a computer to compose music based on rules of composition, but the computer cannot judge the quality of the resultant music. It must be clearly understood that computers are machines which can be programmed to follow instructions: they don't have their own priorities and judgments. Computers are machines which can help mankind in many ways; but they do not threaten us.
Being obedient without exercising 'commonsense' can be very annoying and unproductive. This is illustrated by the experience of a Colonel who sent his obedient peon to a post office with the order "Go to the Post Office and buy ten 25 paisa stamps". The peon went with the money to the post office and did not return for a long time. The Colonel got worried and went in search of him to the post office and found him standing there with the stamps in his hand. When the Colonel became angry and asked him why he was standing there pat came the reply that he was ordered to buy ten 25 paisa stamps but not ordered to return with them!
A consequence of the uncritical acceptance of orders by a computer is the need to give extensive detailed and correct instructions for solving problems. This can be quite challenging.

Q2 (b) How can you represent character in computers explain?

Answer



REPRESENTATION OF CHARACTERS IN COMPUTERS
Physical devices used to store and process data in computers are two-state devices. A switch for example is a two-sill.: device; it can be either ON or OFF. Very reliable recording and reading on a magnetic surface is achieved when the surface is magnetized in either one of two opposite directions. The two states in this case are magnetic field aligned left to right or right to left . Electronic devices such as transistors used in computers function most reliably operated as switches. That is either in a conducting mode or in a non-conducting mode. Thus all data to be stored and processed in computers are transformed or coded as strings of two symbols .One symbol to represent each state. The two symbols normally used are 0 and I. These are known as bits abbreviation for binary digits.
There are 4 unique combinations of two bits namely:
» (00) 01 10 II
There are 2x 2 x 2=8 unique combinations or strings of 3 bits each and they are:

000 001 0I0 0II 100 I0I II0 III
Each unique string of bits may be used to represent or code a symbol. in order to code the 26 capital letters of English at least 26 unique strings of bits are needed. With 4 bits there are only I6 (2 x 2 x2¤2¤ I6)unique strings of 4 bits and 4 bits are not sufficient. Five bits however are sufiicieritas32 (2¤2¤ 2 x 2 ¤2¤32) strings of 5 bits each can be formed. Twenty six out of these 32 strings of 5 bits may be picked to code the 26 letters as illustrated in Table.
Information processing using computers requires processing of only the 26 capital English letters but also the 26 small English letters. I0 digits and around 32 other characters, such as punctuation marks, arithmetic operator symbols, parentheses etc. Total number of 1 character to be coded is thus: 26+26+I0+32¤94. With strings of 6 bits each it is possible to code only 2* ¤ 64 characters. Thus 6 bits are insufficient for coding.

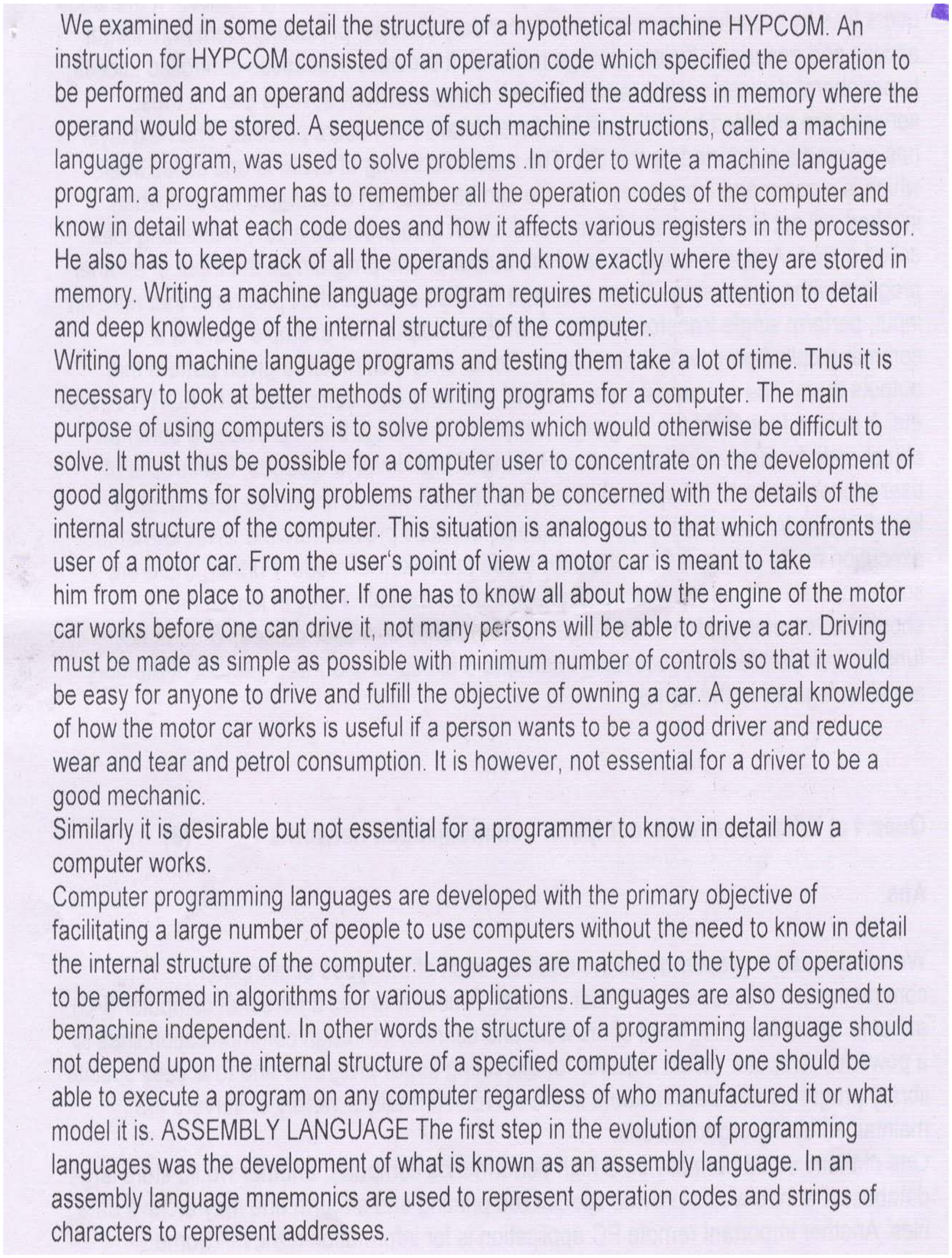
Bit string	*Letter*	*Bit string*	*Letter*
0 0 0 0 0	A	1 0 0 0 0	Q
0 0 0 0 1	B	1 0 0 0 1	R
0 0 0 1 0	C	1 0 0 1 0	S
0 0 0 1 1	D	1 0 0 1 1	T
0 0 1 0 0	E	1 0 1 0 0	U
0 0 1 0 1	F	1 0 1 0 1	V
0 0 1 1 0	G	1 0 1 1 0	W
0 0 1 1 1	H	1 0 1 1 1	X
0 1 0 0 0	I	1 1 0 0 0	Y
0 1 0 0 1	J	1 1 0 0 1	Z
0 1 0 1 0	K	1 1 0 1 0	Not used
0 1 0 1 1	L	1 1 0 1 1	
0 1 1 0 0	M	1 1 1 0 0	
0 1 1 0 1	N	1 1 1 0 1	
0 1 1 1 0	O	1 1 1 1 0	
0 1 1 1 1	P	1 1 1 1 1	

If we use strings of 7 bits each we will have 2 raise to 8 = 128 unique strings and can thus code up to 128 characters. Strings of 7 bits each are thus quite sufficient to code 94 characters. Coding of characters has been standardized to facilitate exchange of recorded data between computers. The most popular standard is known as ASCII (American Standard
Code for Information Interchange) .This uses 7 bits to code each character. Besides codes for characters, in this standard, codes are defined to convey information such as end of line, end of page, etc., to the computer. These codes are said to be control characters which are not printable. Table gives the ASCII code for both printable and non-printable (control) characters. Columns 1 and 2 are non-printable codes. The entry CR, for example Indicates carriage return (or end of line) control character. The most significant bits of the code are
given in Table as column headings and the least significant bits of the code are row headings. Thus the code for A. for example, is identified from the table by finding the column and row bits. The column gives bits 100 as bits b6 b5 b4 and the row gives bits 0001
b3 b2 b1 b0
Thus the code for A is:
b6 b5 b4 b3 b2 b1 b0
1 0 0 0 0 0 1

The internal coded representation of the string RAMA J is:
1010010 I000000I 1001101 1000001 0100000 1001010
R A M A SPACE J
Observe that the blank between RAMA and J also needs a code. This code is essential to leave a blank between RAMA and J when the string is printed.
In addition to ASCII another code known as ISCII (Indian Standard Code for Information Interchange) has been standardized by the Indian Standards Organization.

Q3 (a) Why do we need programming languages?

Answer



We examined in some detail the structure of a hypothetical machine HYPCOM. An instruction for HYPCOM consisted of an operation code which specified the operation to be performed and an operand address which specified the address in memory where the operand would be stored. A sequence of such machine instructions, called a machine language program, was used to solve problems .In order to write a machine language program. a programmer has to remember all the operation codes of the computer and know in detail what each code does and how it affects various registers in the processor. He also has to keep track of all the operands and know exactly where they are stored in memory. Writing a machine language program requires meticulous attention to detail and deep knowledge of the internal structure of the computer.

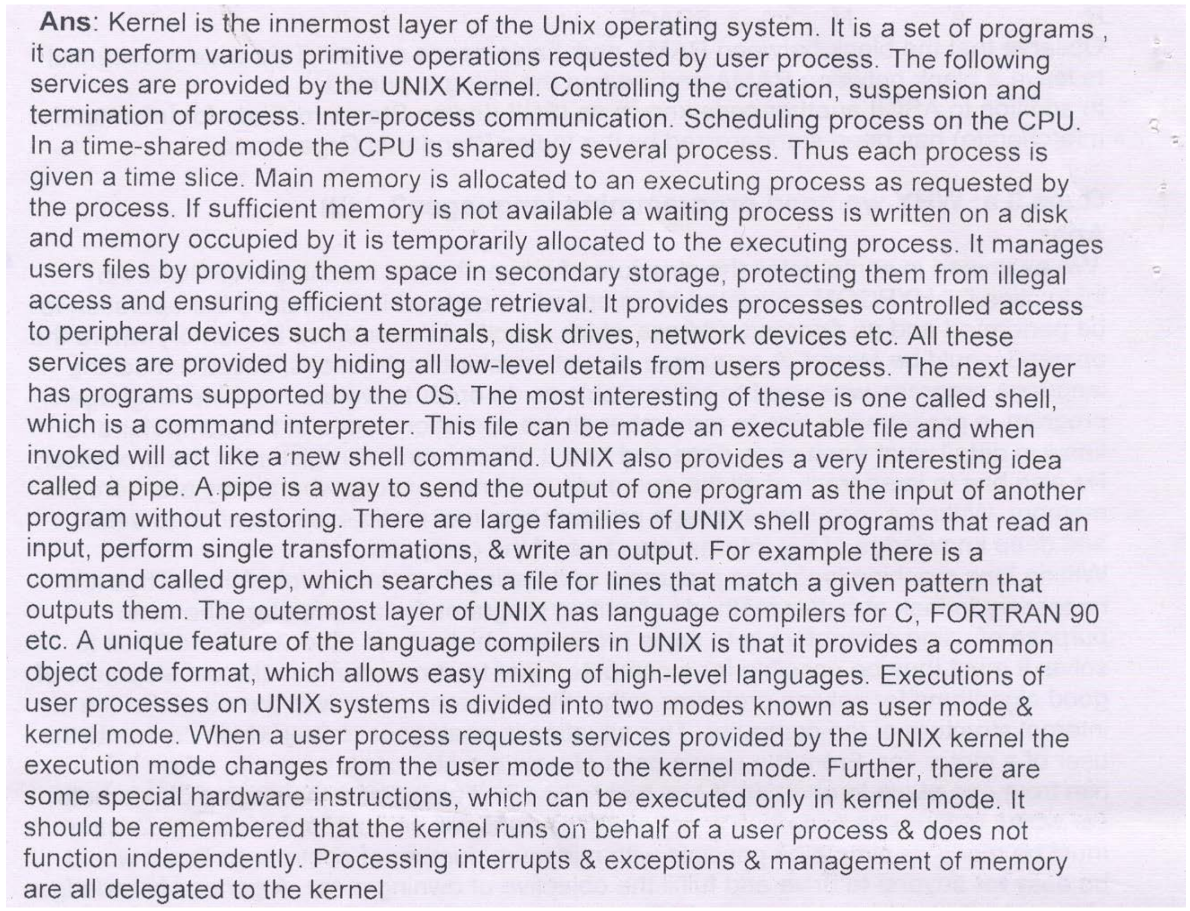
Writing long machine language programs and testing them take a lot of time. Thus it is necessary to look at better methods of writing programs for a computer. The main purpose of using computers is to solve problems which would otherwise be difficult to solve. It must thus be possible for a computer user to concentrate on the development of good algorithms for solving problems rather than be concerned with the details of the internal structure of the computer. This situation is analogous to that which confronts the user of a motor car. From the user's point of view a motor car is meant to take him from one place to another. If one has to know all about how the engine of the motor car works before one can drive it, not many persons will be able to drive a car. Driving must be made as simple as possible with minimum number of controls so that it would be easy for anyone to drive and fulfill the objective of owning a car. A general knowledge of how the motor car works is useful if a person wants to be a good driver and reduce wear and tear and petrol consumption. It is however, not essential for a driver to be a good mechanic.

Similarly it is desirable but not essential for a programmer to know in detail how a computer works.

Computer programming languages are developed with the primary objective of facilitating a large number of people to use computers without the need to know in detail the internal structure of the computer. Languages are matched to the type of operations to be performed in algorithms for various applications. Languages are also designed to bemachine independent. In other words the structure of a programming language should not depend upon the internal structure of a specified computer ideally one should be able to execute a program on any computer regardless of who manufactured it or what model it is. ASSEMBLY LANGUAGE The first step in the evolution of programming languages was the development of what is known as an assembly language. In an assembly language mnemonics are used to represent operation codes and strings of characters to represent addresses.

Q3 (b) Discuss the UNIX operating system in detail.

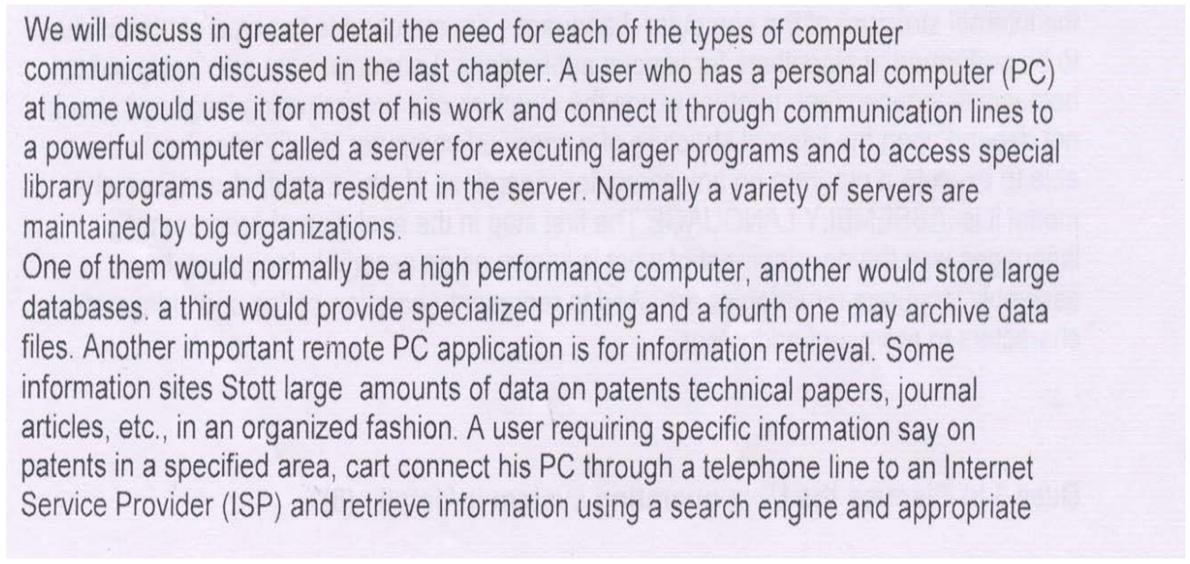
Answer



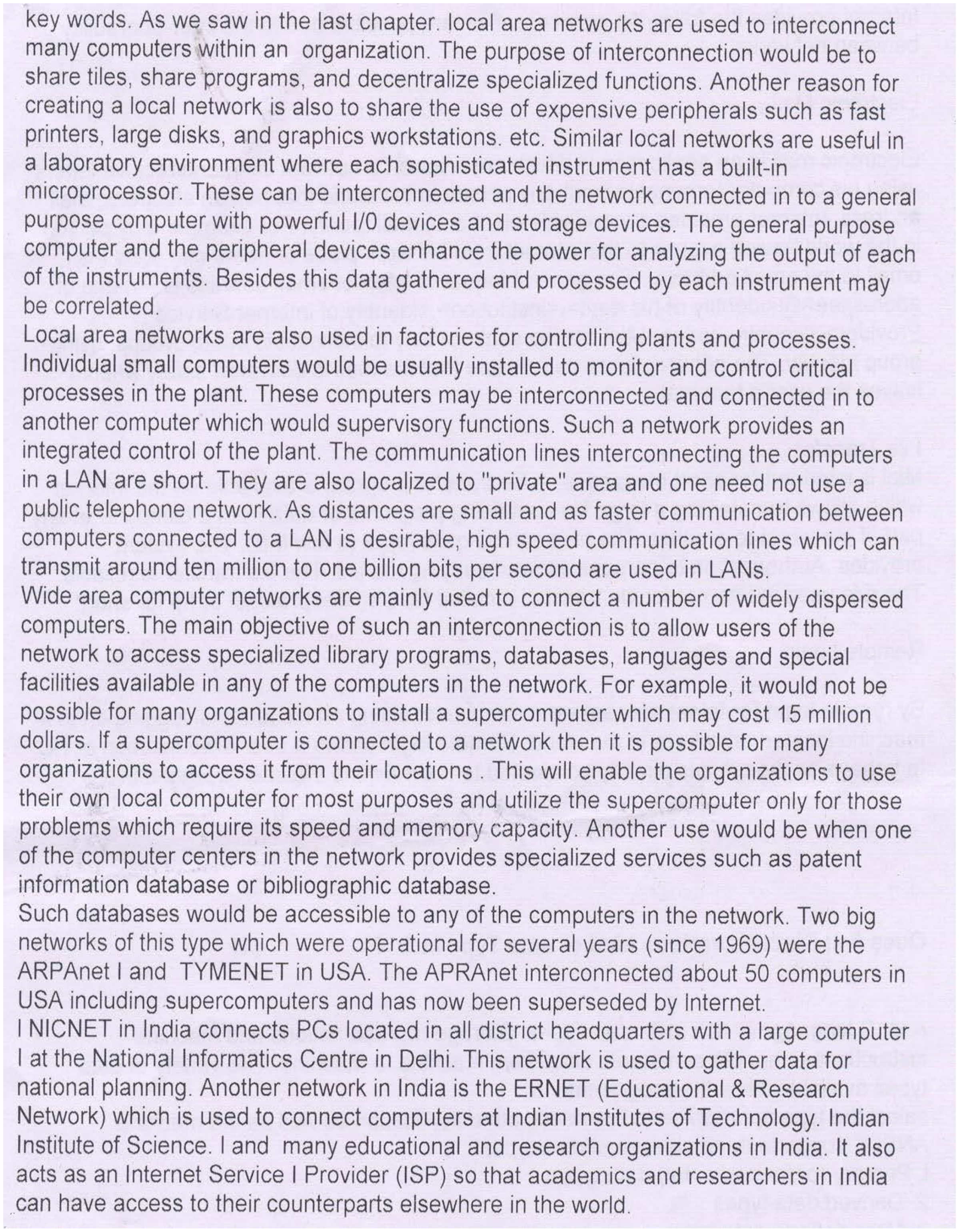
Ans: Kernel is the innermost layer of the Unix operating system. It is a set of programs , it can perform various primitive operations requested by user process. The following services are provided by the UNIX Kernel. Controlling the creation, suspension and termination of process. Inter-process communication. Scheduling process on the CPU. In a time-shared mode the CPU is shared by several process. Thus each process is given a time slice. Main memory is allocated to an executing process as requested by the process. If sufficient memory is not available a waiting process is written on a disk and memory occupied by it is temporarily allocated to the executing process. It manages users files by providing them space in secondary storage, protecting them from illegal access and ensuring efficient storage retrieval. It provides processes controlled access to peripheral devices such a terminals, disk drives, network devices etc. All these services are provided by hiding all low-level details from users process. The next layer has programs supported by the OS. The most interesting of these is one called shell, which is a command interpreter. This file can be made an executable file and when invoked will act like a new shell command. UNIX also provides a very interesting idea called a pipe. A pipe is a way to send the output of one program as the input of another program without restoring. There are large families of UNIX shell programs that read an input, perform single transformations, & write an output. For example there is a command called grep, which searches a file for lines that match a given pattern that outputs them. The outermost layer of UNIX has language compilers for C, FORTRAN 90 etc. A unique feature of the language compilers in UNIX is that it provides a common object code format, which allows easy mixing of high-level languages. Executions of user processes on UNIX systems is divided into two modes known as user mode & kernel mode. When a user process requests services provided by the UNIX kernel the execution mode changes from the user mode to the kernel mode. Further, there are some special hardware instructions, which can be executed only in kernel mode. It should be remembered that the kernel runs on behalf of a user process & does not function independently. Processing interrupts & exceptions & management of memory are all delegated to the kernel

Q4 (a) What is need for computer communication networks?

Answer



We will discuss in greater detail the need for each of the types of computer communication discussed in the last chapter. A user who has a personal computer (PC) at home would use it for most of his work and connect it through communication lines to a powerful computer called a server for executing larger programs and to access special library programs and data resident in the server. Normally a variety of servers are maintained by big organizations.
One of them would normally be a high performance computer, another would store large databases. a third would provide specialized printing and a fourth one may archive data files. Another important remote PC application is for information retrieval. Some information sites Stott large amounts of data on patents technical papers, journal articles, etc., in an organized fashion. A user requiring specific information say on patents in a specified area, cart connect his PC through a telephone line to an Internet Service Provider (ISP) and retrieve information using a search engine and appropriate



key words. As we saw in the last chapter. local area networks are used to interconnect many computers within an organization. The purpose of interconnection would be to share tiles, share programs, and decentralize specialized functions. Another reason for creating a local network is also to share the use of expensive peripherals such as fast printers, large disks, and graphics workstations. etc. Similar local networks are useful in a laboratory environment where each sophisticated instrument has a built-in microprocessor. These can be interconnected and the network connected in to a general purpose computer with powerful I/0 devices and storage devices. The general purpose computer and the peripheral devices enhance the power for analyzing the output of each of the instruments. Besides this data gathered and processed by each instrument may be correlated.

Local area networks are also used in factories for controlling plants and processes. Individual small computers would be usually installed to monitor and control critical processes in the plant. These computers may be interconnected and connected in to another computer which would supervisory functions. Such a network provides an integrated control of the plant. The communication lines interconnecting the computers in a LAN are short. They are also localized to "private" area and one need not use a public telephone network. As distances are small and as faster communication between computers connected to a LAN is desirable, high speed communication lines which can transmit around ten million to one billion bits per second are used in LANs.

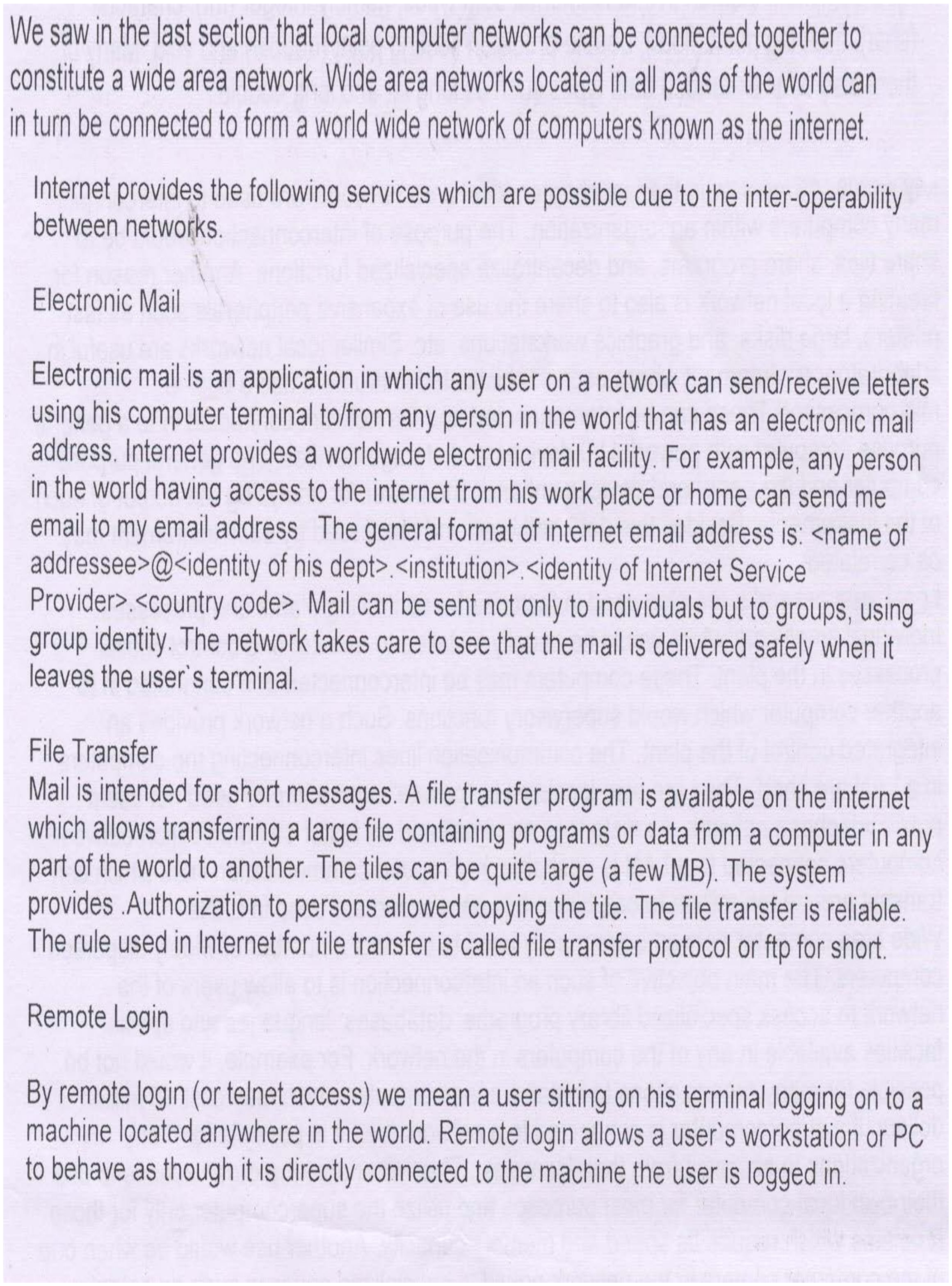
Wide area computer networks are mainly used to connect a number of widely dispersed computers. The main objective of such an interconnection is to allow users of the network to access specialized library programs, databases, languages and special facilities available in any of the computers in the network. For example, it would not be possible for many organizations to install a supercomputer which may cost 15 million dollars. If a supercomputer is connected to a network then it is possible for many organizations to access it from their locations . This will enable the organizations to use their own local computer for most purposes and utilize the supercomputer only for those problems which require its speed and memory capacity. Another use would be when one of the computer centers in the network provides specialized services such as patent information database or bibliographic database.

Such databases would be accessible to any of the computers in the network. Two big networks of this type which were operational for several years (since 1969) were the ARPAnet I and TYMENET in USA. The APRAnet interconnected about 50 computers in USA including supercomputers and has now been superseded by Internet.

I NICNET in India connects PCs located in all district headquarters with a large computer I at the National Informatics Centre in Delhi. This network is used to gather data for national planning. Another network in India is the ERNET (Educational & Research Network) which is used to connect computers at Indian Institutes of Technology. Indian Institute of Science. I and many educational and research organizations in India. It also acts as an Internet Service I Provider (ISP) so that academics and researchers in India can have access to their counterparts elsewhere in the world.

Q4 (b) Write a note on internet and world wide web.

Answer:


We saw in the last section that local computer networks can be connected together to constitute a wide area network. Wide area networks located in all pans of the world can in turn be connected to form a world wide network of computers known as the internet.

Internet provides the following services which are possible due to the inter-operability between networks.

Electronic Mail

Electronic mail is an application in which any user on a network can send/receive letters using his computer terminal to/from any person in the world that has an electronic mail address. Internet provides a worldwide electronic mail facility. For example, any person in the world having access to the internet from his work place or home can send me email to my email address The general format of internet email address is: <name of addressee>@<identity of his dept>.<institution>.<identity of Internet Service Provider>.<country code>. Mail can be sent not only to individuals but to groups, using group identity. The network takes care to see that the mail is delivered safely when it leaves the user`s terminal.

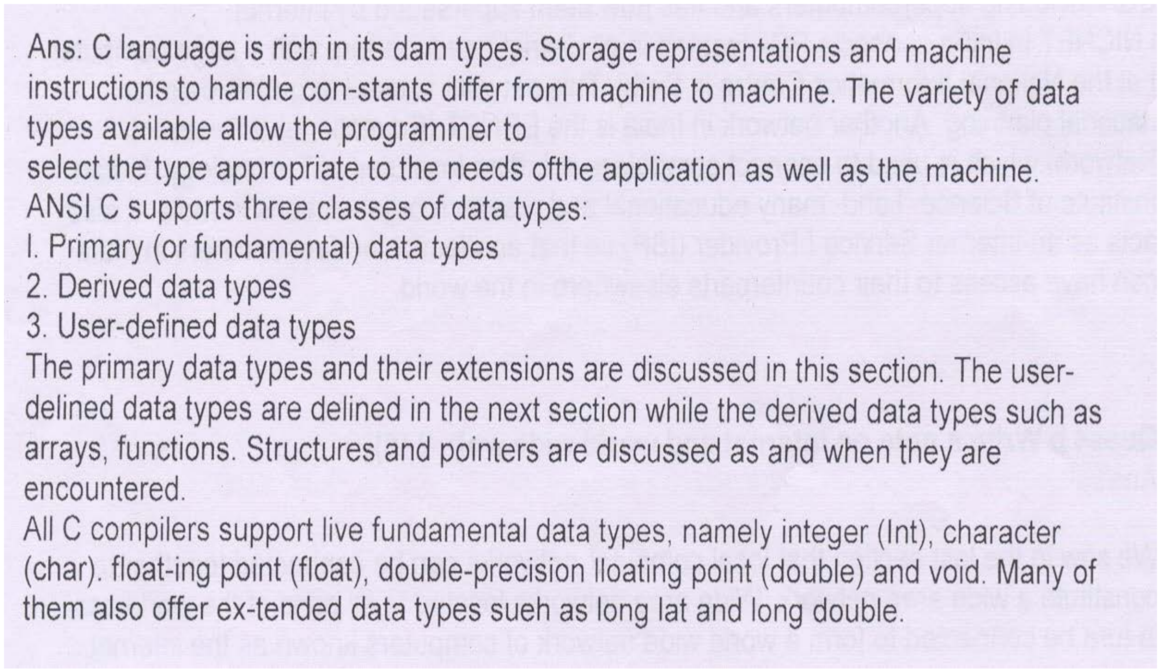
File Transfer
Mail is intended for short messages. A file transfer program is available on the internet which allows transferring a large file containing programs or data from a computer in any part of the world to another. The tiles can be quite large (a few MB). The system provides .Authorization to persons allowed copying the tile. The file transfer is reliable. The rule used in Internet for tile transfer is called file transfer protocol or ftp for short.

Remote Login

By remote login (or telnet access) we mean a user sitting on his terminal logging on to a machine located anywhere in the world. Remote login allows a user`s workstation or PC to behave as though it is directly connected to the machine the user is logged in.

Q5 (a) Discuss various kind of data Type in C.

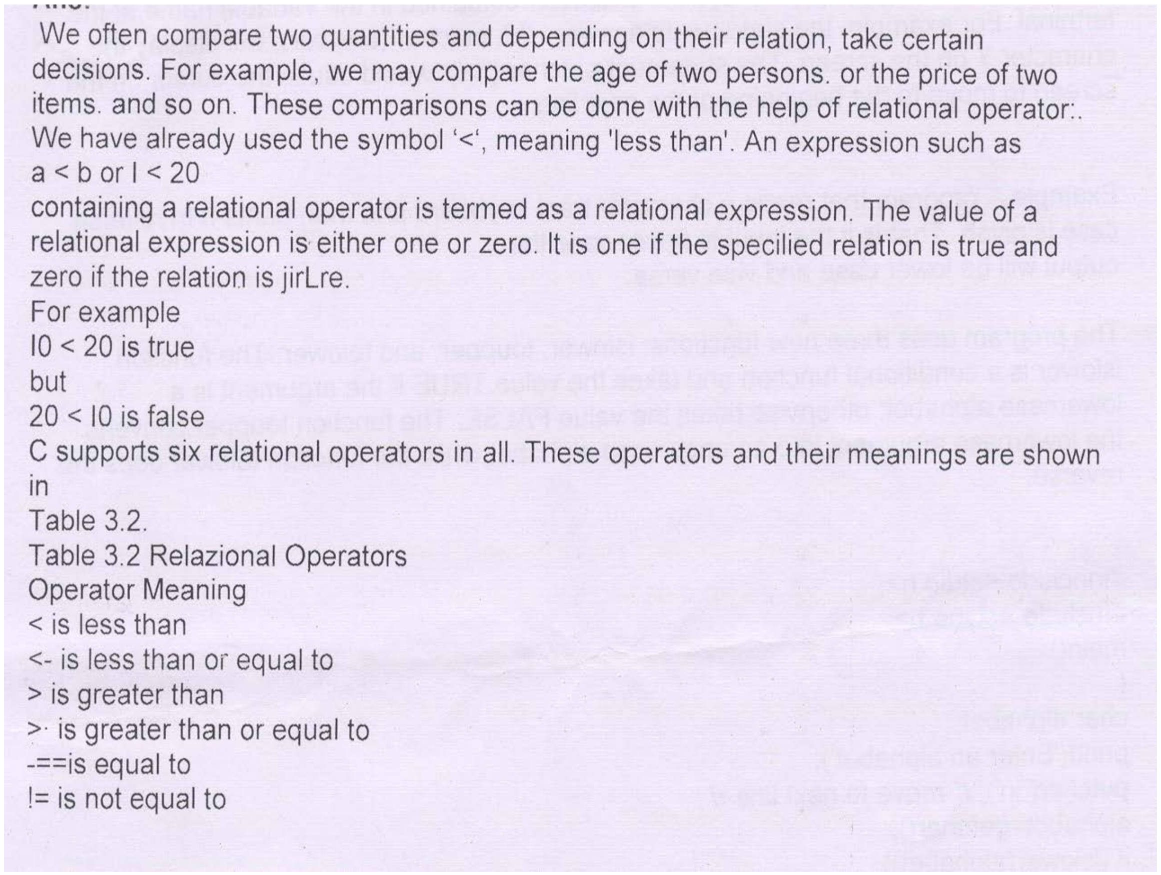
Answer:



Ans: C language is rich in its dam types. Storage representations and machine instructions to handle con-stants differ from machine to machine. The variety of data types available allow the programmer to
select the type appropriate to the needs ofthe application as well as the machine.
ANSI C supports three classes of data types:
I. Primary (or fundamental) data types
2. Derived data types
3. User-defined data types
The primary data types and their extensions are discussed in this section. The user-delined data types are delined in the next section while the derived data types such as arrays, functions. Structures and pointers are discussed as and when they are encountered.
All C compilers support live fundamental data types, namely integer (Int), character (char). float-ing point (float), double-precision lloating point (double) and void. Many of them also offer ex-tended data types sueh as long lat and long double.

Q5 (b) What do you mean by Relational operators?

Answer

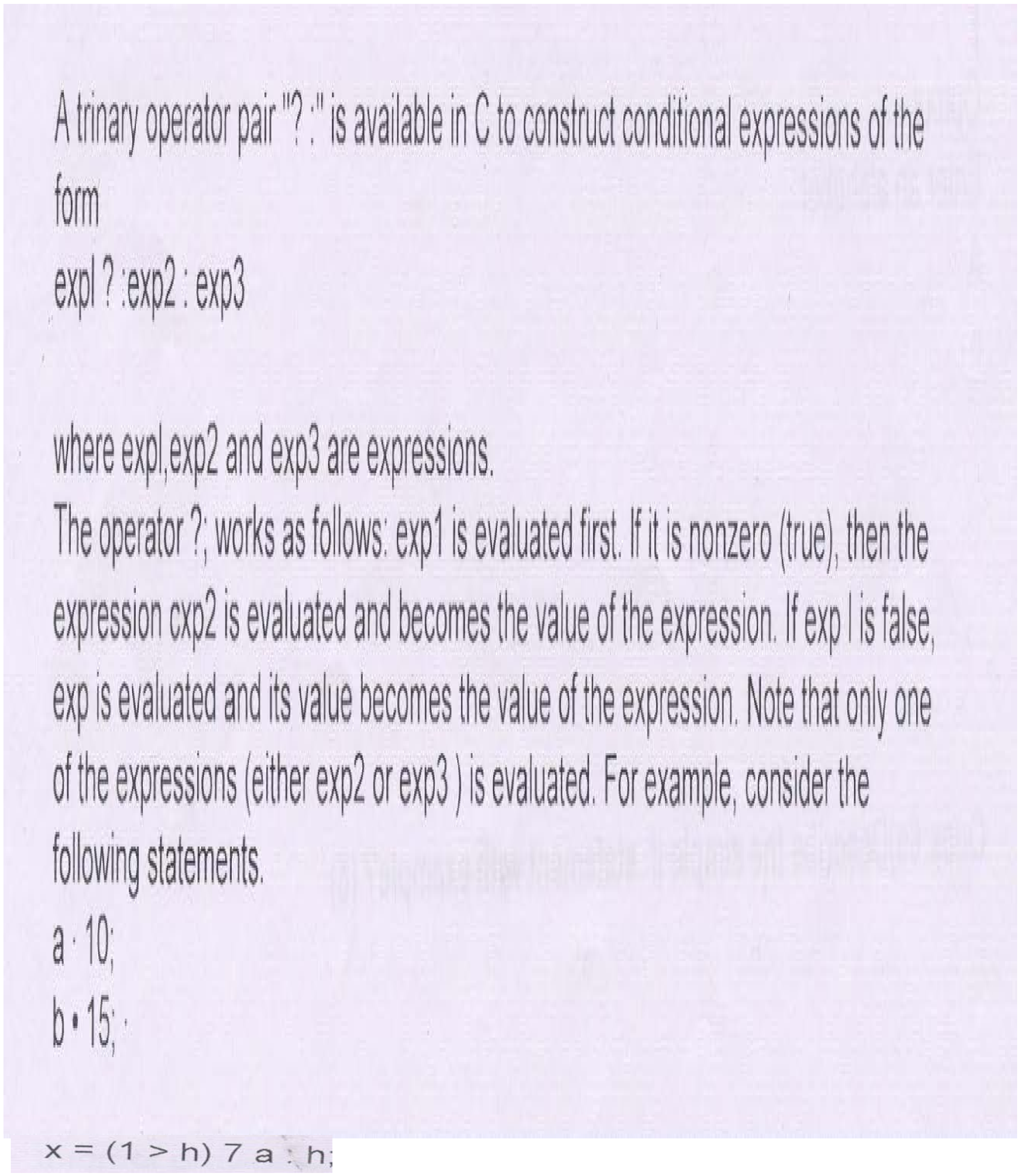


We often compare two quantities and depending on their relation, take certain decisions. For example, we may compare the age of two persons. or the price of two items. and so on. These comparisons can be done with the help of relational operator:.
We have already used the symbol '<', meaning 'less than'. An expression such as
a < b or I < 20
containing a relational operator is termed as a relational expression. The value of a relational expression is either one or zero. It is one if the specilied relation is true and zero if the relation is jirLre.
For example
I0 < 20 is true
but
20 < I0 is false
C supports six relational operators in all. These operators and their meanings are shown in
Table 3.2.
Table 3.2 Relazional Operators

Operator	Meaning
<	is less than
<-	is less than or equal to
>	is greater than
>·	is greater than or equal to
-==	is equal to
!=	is not equal to

Q5 (c) Discuss the conditional operators.

Answer:



A trinary operator pair "? :" is available in C to construct conditional expressions of the form

expl ? :exp2 : exp3

where expl,exp2 and exp3 are expressions.

The operator ?; works as follows: exp1 is evaluated first. If it is nonzero (true), then the expression cxp2 is evaluated and becomes the value of the expression. If exp l is false, exp is evaluated and its value becomes the value of the expression. Note that only one of the expressions (either exp2 or exp3) is evaluated. For example, consider the following statements.

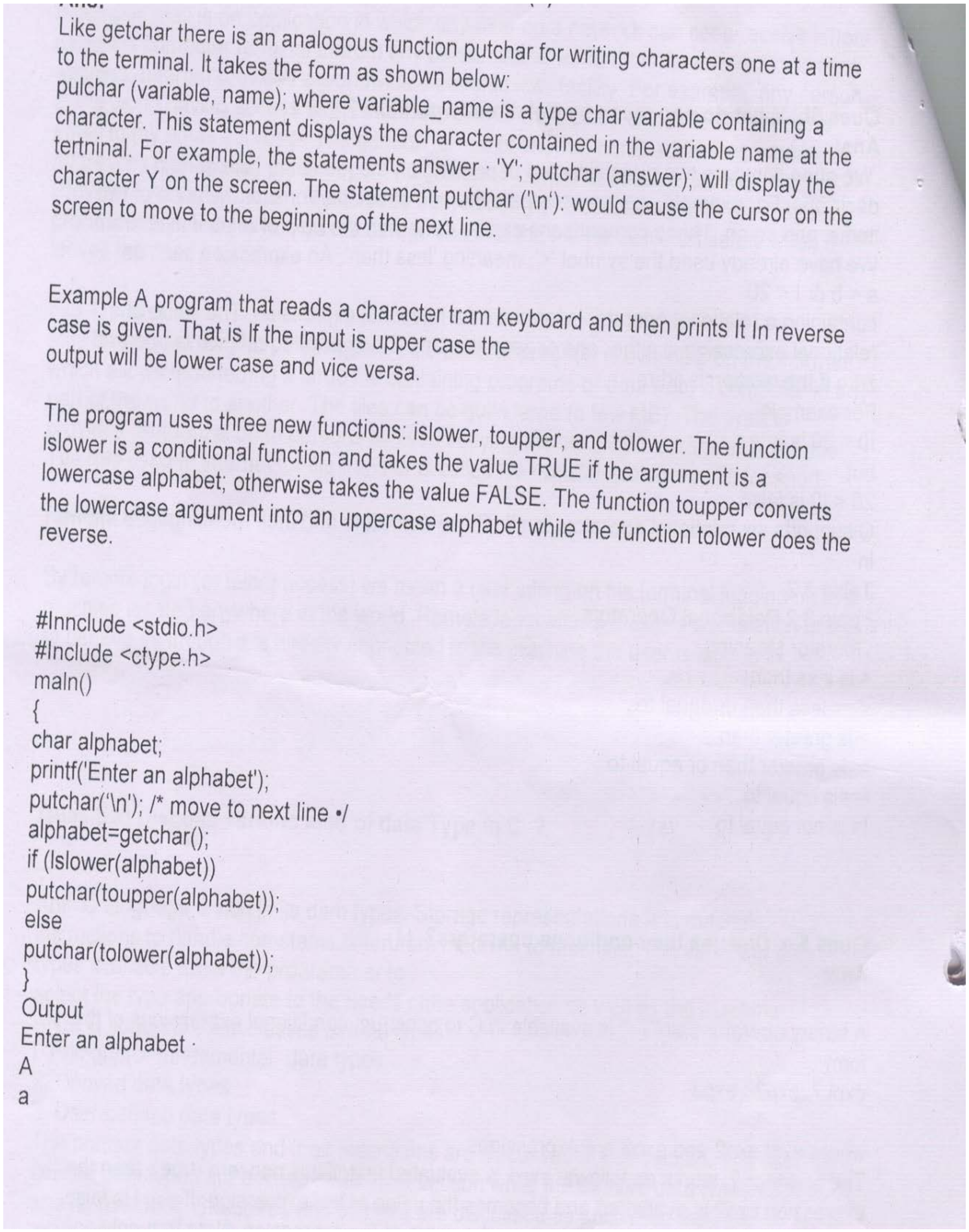
a · 10;

b • 15;

x = (1 > h) 7 a : h;

Q5 (d) Discuss WRITING A CHARACTER ?

Answer



Like getchar there is an analogous function putchar for writing characters one at a time to the terminal. It takes the form as shown below:
pulchar (variable, name); where variable_name is a type char variable containing a character. This statement displays the character contained in the variable name at the tertninal. For example, the statements answer · 'Y'; putchar (answer); will display the character Y on the screen. The statement putchar ('\n'): would cause the cursor on the screen to move to the beginning of the next line.

Example A program that reads a character tram keyboard and then prints It In reverse case is given. That is If the input is upper case the output will be lower case and vice versa.

The program uses three new functions: islower, toupper, and tolower. The function islower is a conditional function and takes the value TRUE if the argument is a lowercase alphabet; otherwise takes the value FALSE. The function toupper converts the lowercase argument into an uppercase alphabet while the function tolower does the reverse.

```
#Innclude <stdio.h>
#Include <ctype.h>
maln()
{
char alphabet;
printf('Enter an alphabet');
putchar('\n'); /* move to next line •/
alphabet=getchar();
if (lslower(alphabet))
putchar(toupper(alphabet));
else
putchar(tolower(alphabet));
}
```

Output

```
Enter an alphabet ·
A
a
```

Q6 (a) Describe the simple if statement with example.

Answer

The simple if statement is a powerful decision-making statement and is used to control the (low of execution of statements. It is basically a two·way decision statement and is used in conjunction with an
expression. It takes the following

It allows the computer to evaluate the expression first and then, depending on whether the value of the expression (relation or condition) is 'true' (or non-zero) or 'false' (zero), it transfers the control to a particular statement. This point of program has two paths to follow, one for the true condition and the other for the false condition
Some examples of decision making, using If statements are:
I. if (bank balance is zero)
borrow money
2. if (room is dark)
put on lights
3. II` (code is l)
person is male
4. If (age is more than 55)
person is retired
The If statement may be implemented in different fortrans depending on the complexity of conditions to be tested. The different forms are:
I. Simple If statement
2. if ..... else statement
3. Nested if .... else statement
4. else if ladder.
We shall discuss each one of them in the next few sections.

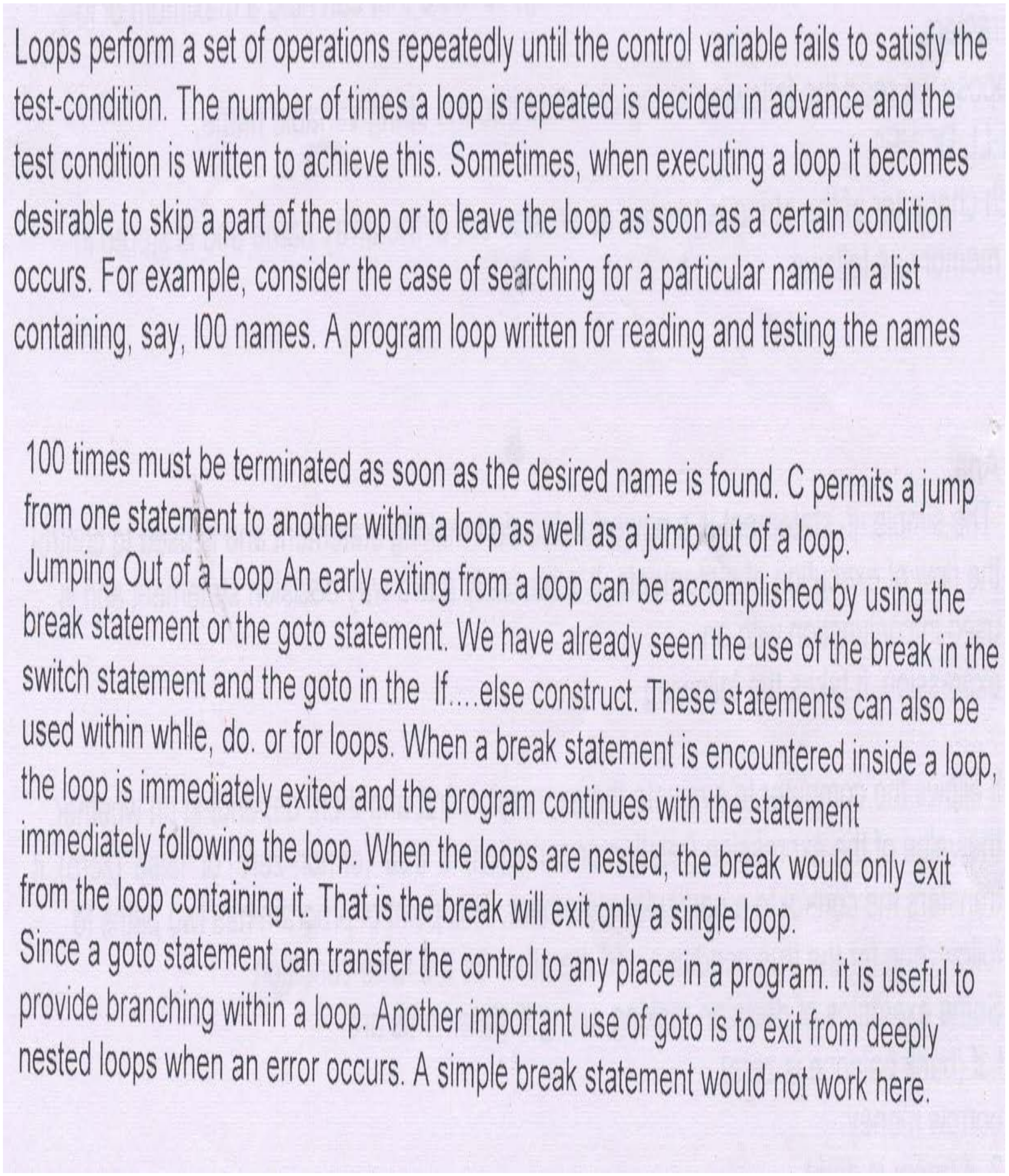
SIMPLE IF STATEMENT

The general form of a simple If statement is

```
If (test expression)
(
statement-block;
)
statement-x;
```

Q6 (b) What do you mean by jumps in loops?

Answer

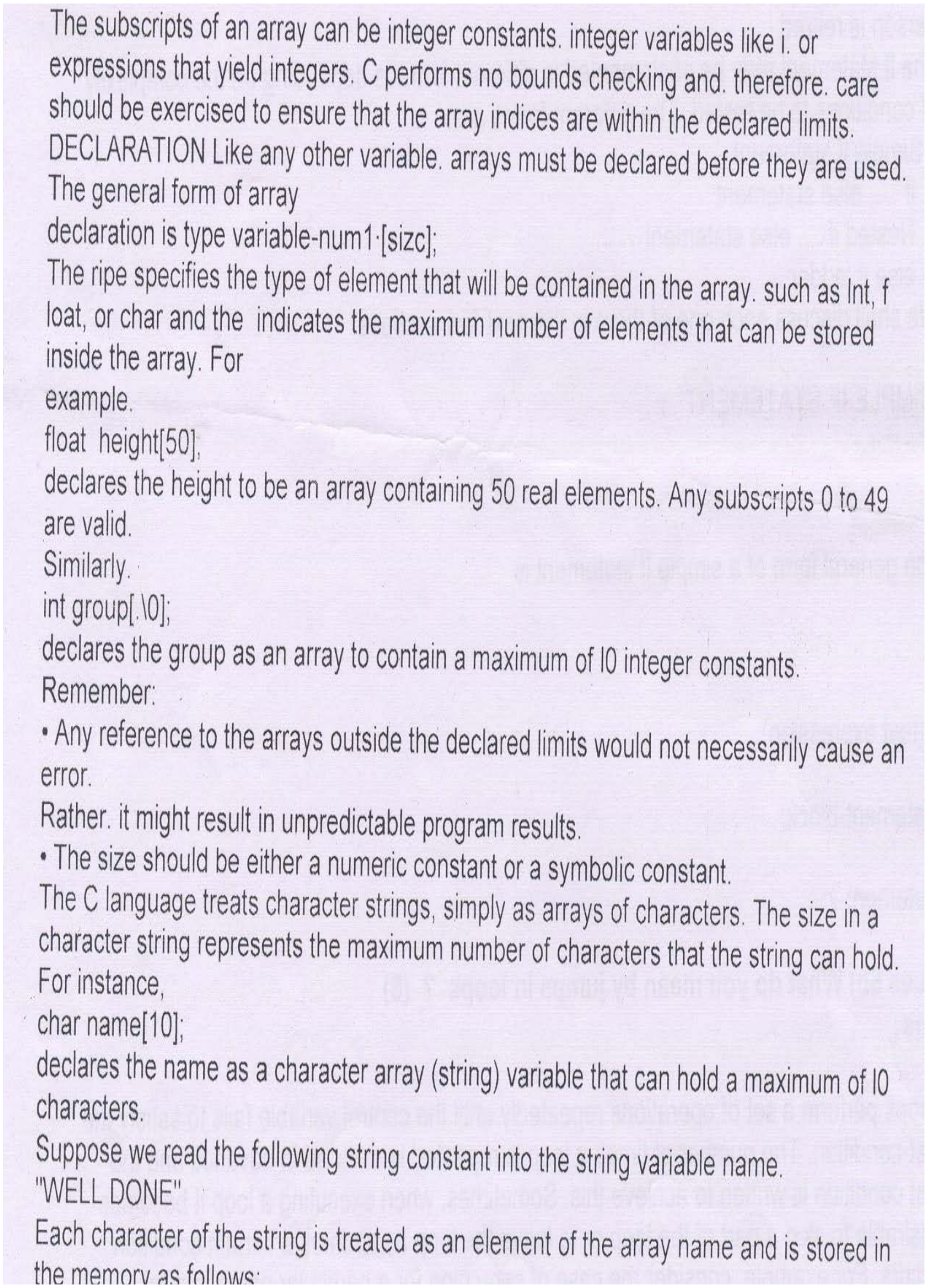


Loops perform a set of operations repeatedly until the control variable fails to satisfy the test-condition. The number of times a loop is repeated is decided in advance and the test condition is written to achieve this. Sometimes, when executing a loop it becomes desirable to skip a part of the loop or to leave the loop as soon as a certain condition occurs. For example, consider the case of searching for a particular name in a list containing, say, I00 names. A program loop written for reading and testing the names

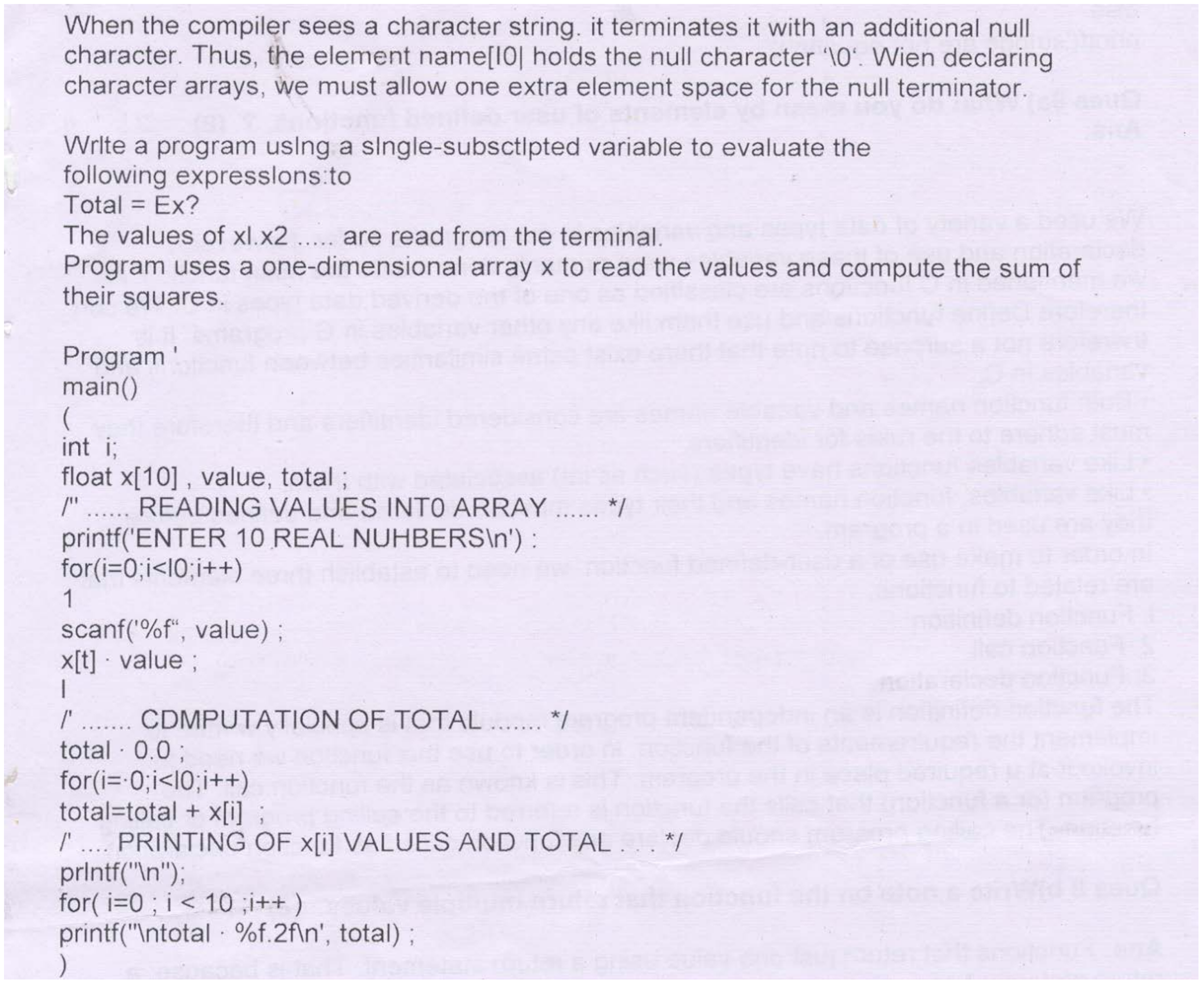
100 times must be terminated as soon as the desired name is found. C permits a jump from one statement to another within a loop as well as a jump out of a loop.
Jumping Out of a Loop An early exiting from a loop can be accomplished by using the break statement or the goto statement. We have already seen the use of the break in the switch statement and the goto in the If....else construct. These statements can also be used within whlle, do. or for loops. When a break statement is encountered inside a loop, the loop is immediately exited and the program continues with the statement immediately following the loop. When the loops are nested, the break would only exit from the loop containing it. That is the break will exit only a single loop.
Since a goto statement can transfer the control to any place in a program. it is useful to provide branching within a loop. Another important use of goto is to exit from deeply nested loops when an error occurs. A simple break statement would not work here.

Q7 (a) Explain the declaration of arrays with example.

Answer



The subscripts of an array can be integer constants. integer variables like i. or expressions that yield integers. C performs no bounds checking and. therefore. care should be exercised to ensure that the array indices are within the declared limits.
DECLARATION Like any other variable. arrays must be declared before they are used. The general form of array
declaration is type variable-num1·[sizc];
The ripe specifies the type of element that will be contained in the array. such as Int, f loat, or char and the indicates the maximum number of elements that can be stored inside the array. For
example.
float height[50];
declares the height to be an array containing 50 real elements. Any subscripts 0 to 49 are valid.
Similarly.
int group[.\0];
declares the group as an array to contain a maximum of I0 integer constants.
Remember:
• Any reference to the arrays outside the declared limits would not necessarily cause an error.
Rather. it might result in unpredictable program results.
• The size should be either a numeric constant or a symbolic constant.
The C language treats character strings, simply as arrays of characters. The size in a character string represents the maximum number of characters that the string can hold.
For instance,
char name[10];
declares the name as a character array (string) variable that can hold a maximum of I0 characters.
Suppose we read the following string constant into the string variable name.
"WELL DONE"
Each character of the string is treated as an element of the array name and is stored in the memory as follows:

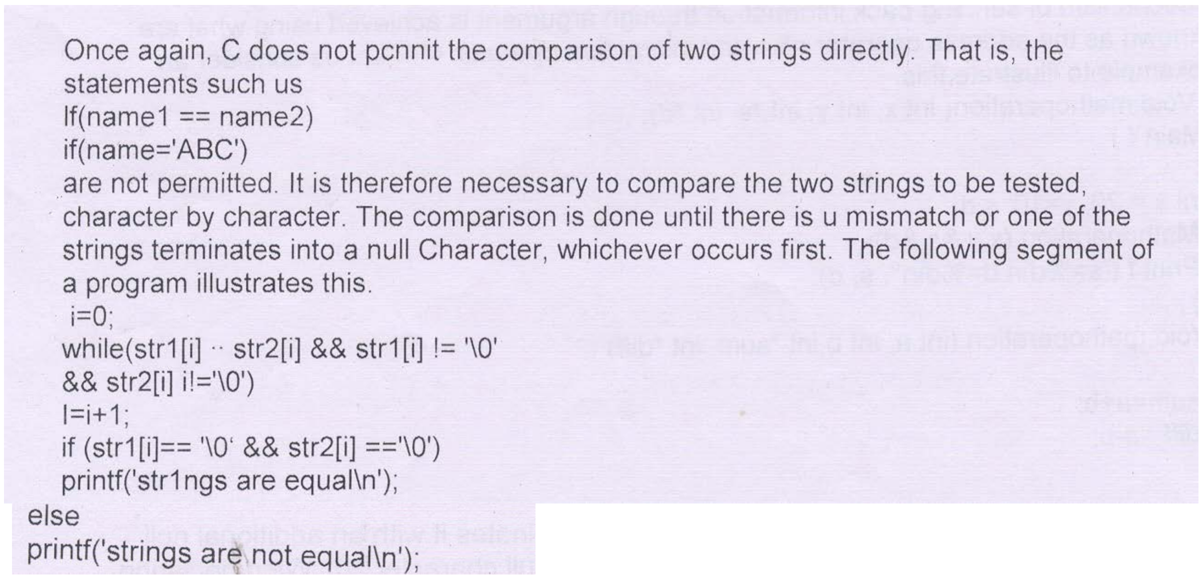

When the compiler sees a character string. it terminates it with an additional null character. Thus, the element name[I0| holds the null character '\0'. Wien declaring character arrays, we must allow one extra element space for the null terminator.

Wrlte a program uslng a slngle-subsctlpted variable to evaluate the following expresslons:to
Total = Ex?
The values of xl.x2 ..... are read from the terminal.
Program uses a one-dimensional array x to read the values and compute the sum of their squares.

```
Program '
main()
(
int  i;
float x[10] , value, total ;
/" ...... READING VALUES INT0 ARRAY ...... */
printf('ENTER 10 REAL NUHBERS\n') :
for(i=0;i<l0;i++)
1
scanf('%f", value) ;
x[t] · value ;
I
/' ....... CDMPUTATION OF TOTAL ....... */
total · 0.0 ;
for(i=·0;i<l0;i++)
total=total + x[i] ;
/' .... PRINTING OF x[i] VALUES AND TOTAL . . . */
prlntf("\n");
for( i=0 ; i < 10 ;i++ )
printf("\ntotal · %f.2f\n', total) ;
)
```

Q7 (b) Explain the comparison of two strings.

Answer


Once again, C does not pcnnit the comparison of two strings directly. That is, the statements such us

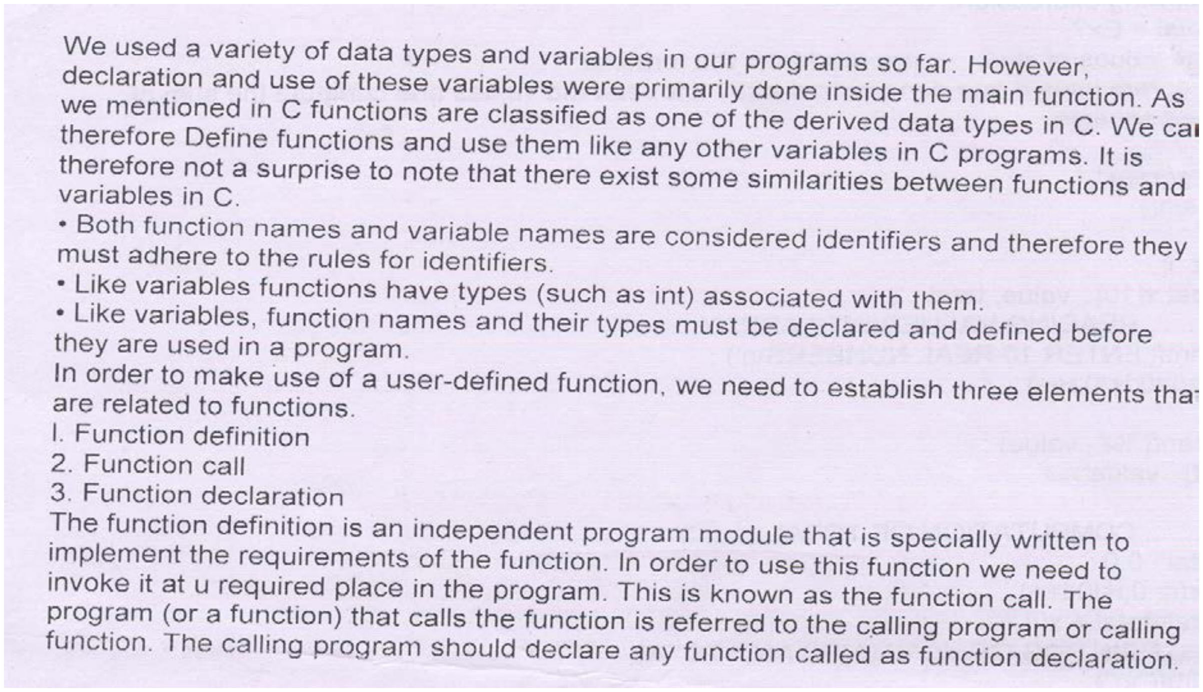
```
If(name1 == name2)
if(name='ABC')
```

are not permitted. It is therefore necessary to compare the two strings to be tested, character by character. The comparison is done until there is u mismatch or one of the strings terminates into a null Character, whichever occurs first. The following segment of a program illustrates this.

```
 i=0;
while(str1[i] ·· str2[i] && str1[i] != '\0'
&& str2[i] i!='\0')
I=i+1;
if (str1[i]== '\0' && str2[i] =='\0')
printf('str1ngs are equal\n');
else
printf('strings are not equal\n');
```

Q8 (a) What do you mean by elements of user defined functions?

Answer



We used a variety of data types and variables in our programs so far. However, declaration and use of these variables were primarily done inside the main function. As we mentioned in C functions are classified as one of the derived data types in C. We ca therefore Define functions and use them like any other variables in C programs. It is therefore not a surprise to note that there exist some similarities between functions and variables in C.

• Both function names and variable names are considered identifiers and therefore they must adhere to the rules for identifiers.

• Like variables functions have types (such as int) associated with them.

• Like variables, function names and their types must be declared and defined before they are used in a program.

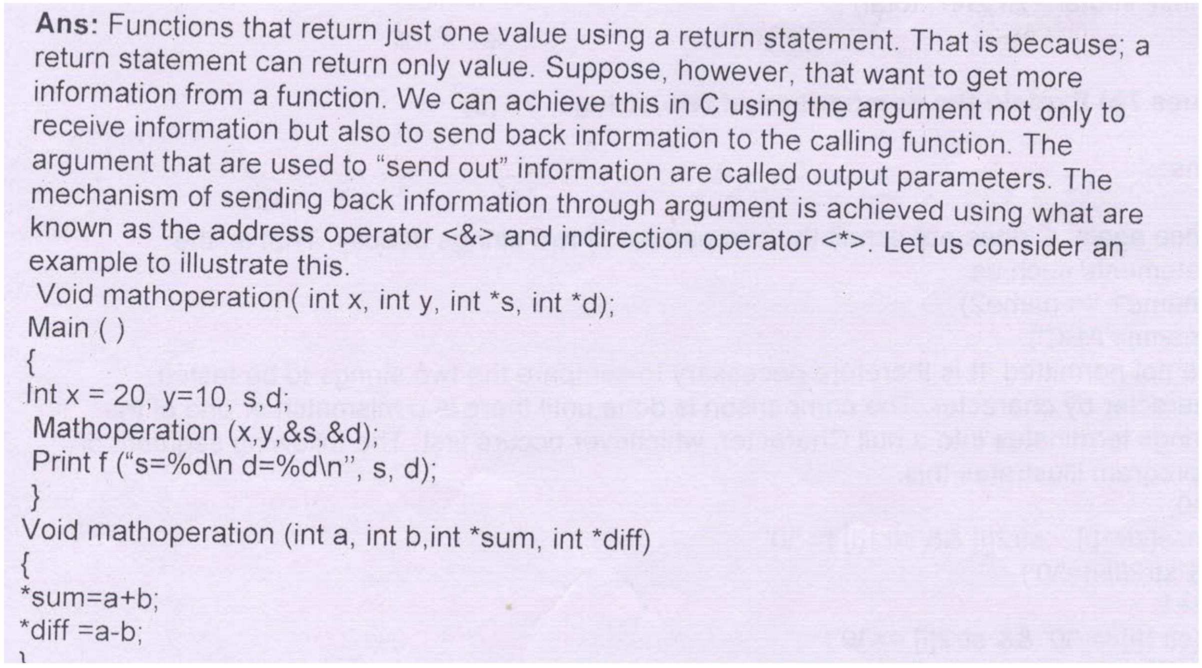
In order to make use of a user-defined function, we need to establish three elements tha are related to functions.

I. Function definition
2. Function call
3. Function declaration

The function definition is an independent program module that is specially written to implement the requirements of the function. In order to use this function we need to invoke it at u required place in the program. This is known as the function call. The program (or a function) that calls the function is referred to the calling program or calling function. The calling program should declare any function called as function declaration.

Q8 (b) Write a note on the function that return multiple values.

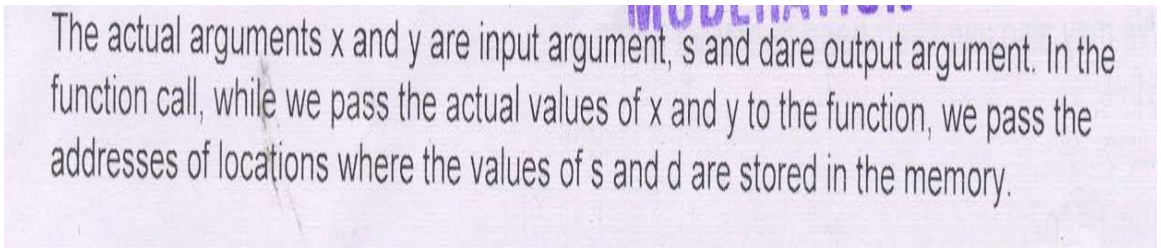
Answer



Ans: Functions that return just one value using a return statement. That is because; a return statement can return only value. Suppose, however, that want to get more information from a function. We can achieve this in C using the argument not only to receive information but also to send back information to the calling function. The argument that are used to "send out" information are called output parameters. The mechanism of sending back information through argument is achieved using what are known as the address operator <&> and indirection operator <*>. Let us consider an example to illustrate this.

```
 Void mathoperation( int x, int y, int *s, int *d);
Main ( )
{
Int x = 20, y=10, s,d;
 Mathoperation (x,y,&s,&d);
 Print f ("s=%d\n d=%d\n", s, d);
 }
Void mathoperation (int a, int b,int *sum, int *diff)
{
*sum=a+b;
*diff =a-b;
}
```

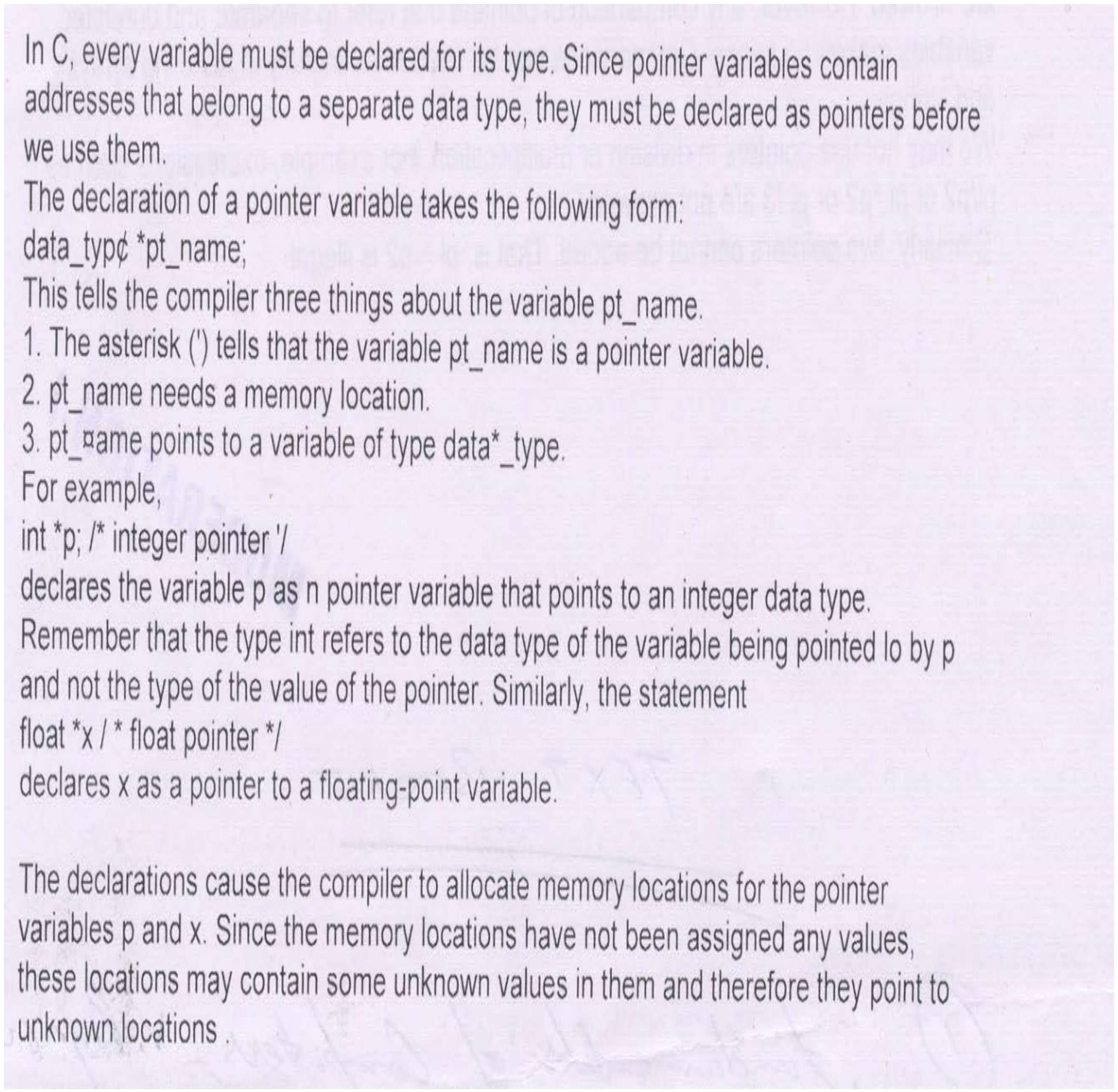
}



The actual arguments x and y are input argument, s and dare output argument. In the function call, while we pass the actual values of x and y to the function, we pass the addresses of locations where the values of s and d are stored in the memory.

Q9 (a) What do you mean by declaring pointer variables?

Answer

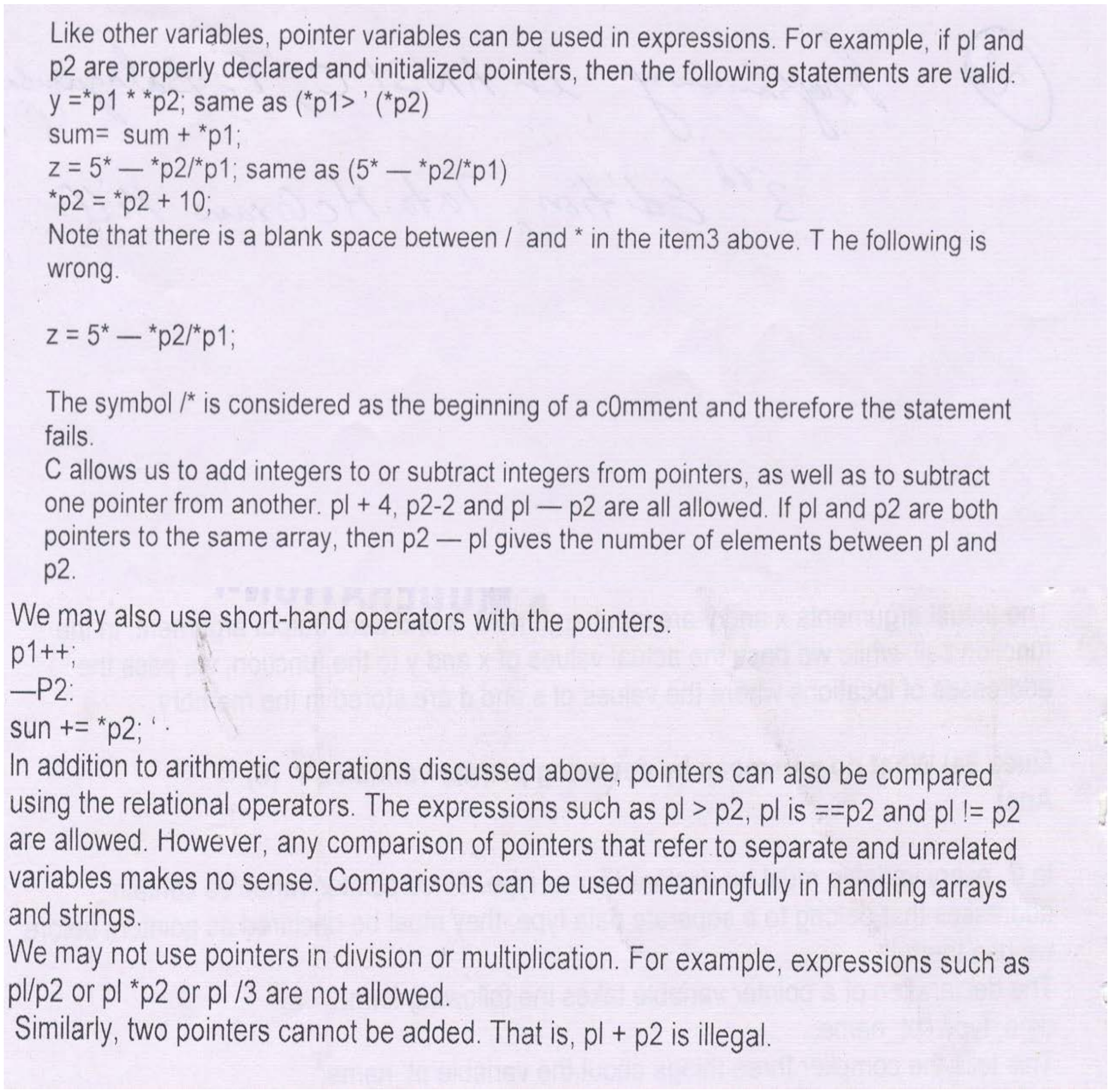


In C, every variable must be declared for its type. Since pointer variables contain addresses that belong to a separate data type, they must be declared as pointers before we use them.
The declaration of a pointer variable takes the following form:
data_typ¢ *pt_name;
This tells the compiler three things about the variable pt_name.
1. The asterisk (') tells that the variable pt_name is a pointer variable.
2. pt_name needs a memory location.
3. pt_¤ame points to a variable of type data*_type.
For example,
int *p; /* integer pointer '/
declares the variable p as n pointer variable that points to an integer data type.
Remember that the type int refers to the data type of the variable being pointed lo by p and not the type of the value of the pointer. Similarly, the statement
float *x / * float pointer */
declares x as a pointer to a floating-point variable.

The declarations cause the compiler to allocate memory locations for the pointer variables p and x. Since the memory locations have not been assigned any values, these locations may contain some unknown values in them and therefore they point to unknown locations .

Q9 (b) Write a note on Pointer expressions with example.

Answer



Like other variables, pointer variables can be used in expressions. For example, if pl and p2 are properly declared and initialized pointers, then the following statements are valid.
y =*p1 * *p2; same as (*p1> ' (*p2)
sum= sum + *p1;
z = 5* — *p2/*p1; same as (5* — *p2/*p1)
*p2 = *p2 + 10;
Note that there is a blank space between / and * in the item3 above. T he following is wrong.

z = 5* — *p2/*p1;

The symbol /* is considered as the beginning of a c0mment and therefore the statement fails.
C allows us to add integers to or subtract integers from pointers, as well as to subtract one pointer from another. pl + 4, p2-2 and pl — p2 are all allowed. If pl and p2 are both pointers to the same array, then p2 — pl gives the number of elements between pl and p2.
We may also use short-hand operators with the pointers.
p1++;
—P2:
sun += *p2; ' ·
In addition to arithmetic operations discussed above, pointers can also be compared using the relational operators. The expressions such as pl > p2, pl is ==p2 and pl != p2 are allowed. However, any comparison of pointers that refer to separate and unrelated variables makes no sense. Comparisons can be used meaningfully in handling arrays and strings.
We may not use pointers in division or multiplication. For example, expressions such as pl/p2 or pl *p2 or pl /3 are not allowed.
Similarly, two pointers cannot be added. That is, pl + p2 is illegal.

Text Books

1. **Fundamentals of Computers, V. Rajaraman, 4th Edition, PHI, 2007**

2. **Programming in ANSI C, E Balagurusamy, 3rd Edition, Tata McGraw Hill**