CHANDIGARH ENGINEERING COLLEGE, LANDRAN (MOHALI)

DEPARTMENT OF APPLIED SCIENCES

PPS (BTPS-101-18) **Branch-ECE/ME Sem-1**

B.Tech -Question Paper-2 Solution
Even Sem -2018-2019- QP

Section-A(10*2= 20 Marks)

1. Write short Notes on:

a) What is the syntax of if –else statement?

Ans:

Syntax of if-else statement:

```
if(condition)
{
//statements to be executed if condition is true
}
else
{//statements to be executed if condition is false}
```

Example:

```
  if(a%2==0)
{
  printf("a is even number");
}
Else
{
  printf("a is odd number");
}
```

b) What pseudocode?

Ans:

"Pseudo" means imitation of false and "code" refers to the instruction written in the programming language. Pseudo code is programming analysis tool that is used for planning program logic.

c) What is a flowchart?

Ans:

A Flowchart is a pictorial representation of an algorithm. It is often used by programmer as a program planning tool for organizing a sequence of step necessary to solve a problem by a computer.

d) What is the difference between primary memory and secondary memory?

Primary memory	Secondary memory
Primary memory is directly accessible by Processor/CPU.	Secondary memory is not directly accessible by the CPU.
Primary memory devices are more expensive than secondary storage devices.	Secondary memory devices are less expensive when compared to primary memory devices.
Examples: RAM, ROM, Cache memory, PROM, EPROM, Registers, etc.	Examples- Pendrive,CD,Floppydisk

e) What is the scope of extern variable?

External variables are also known as global variables. These variables are defined outside the function. These variables are available globally throughout the function execution. extern" keyword is used to declare it.

extern int x = 32;

f) What are the arithmetic operators available in C language?

An arithmetic operator performs mathematical operations such as addition, subtraction and multiplication on numerical values (constants and variables).

Operator	Meaning of Operator
+	addition or unary plus
-	subtraction or unary minus
*	multiplication
/	division
%	remainder after division(modulo division)

g) How many bytes are required for real a[20] statement ?

Ans:

Real is used to denote float variable. So 20 *4=80 bytes are reserved.

h) What are the syntax of printing statements in a programming language?

Ans:

Syntax of PRINTF statement:

Printf('Data to be printed");

If any value has to be printed then

Printf("Format specifier", variablename);

Example:

Printf("%d is an even number",a);

i) What is PRINTF statement?

Ans:PRINTF statement is used to print the text on the screen.

Syntax of PRINTF statement:

```
  Printf('Text to be printed");
```

If any value has to be printed then

```
Printf("Format specifier", variablename);
```

Example:

```
Printf("%d is an even number",a);
```

j) What is a compiler?

Ans:

Compiler is software which translates the entire program into machine code. It generates intermediate object code which requires linking. C, C++ are languages which use compiler.

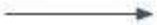
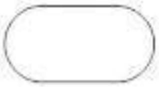
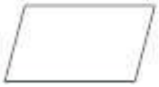
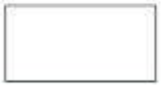
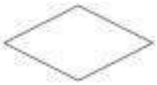
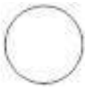
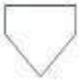
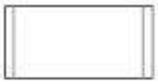
Total 5 – Questions to be attended opting at least two from section B and C (5*8=40 Marks)

Section- B (8 marks Questions)

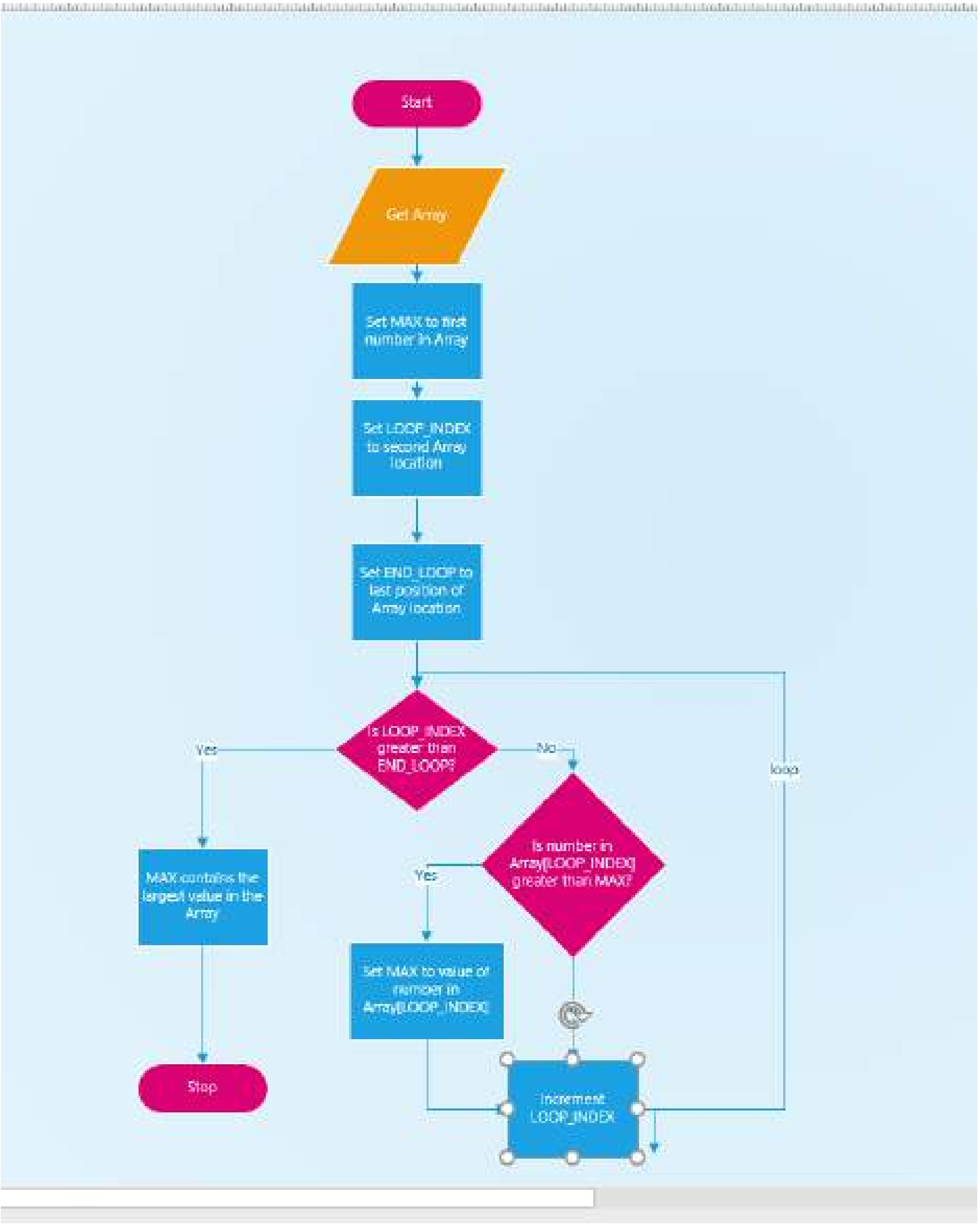
2. What is an algorithm? Differentiate between an algorithm and a flowchart. What are the symbols used in flowchart?

Algorithm means the logic of a program. It is a step-by-step description of how to arrive at a solution of a given problem.

Algorithm	**Flowchart**
1. A method of representing the step-by-step logical procedure for solving a problem.	1. Flow chart is diagrammatic representation of an algorithm. It is constructed using different types of boxes and symbols.
2. It contains step-by-step English descriptions, each step representing a particular operation leading to solution of problem.	2. The flowchart employs a series of blocks and arrows, each of which represents a particular step in an algorithm.
3. These are particularly useful for small problems	3. These are useful for detailed representations of complicated programs.
4. For complex programs, algorithms prove to be Inadequate	4. For complex programs, Flowcharts prove to be adequate

Symbol	Purpose	Description
	Flow line	Used to indicate the flow of logic by connecting symbols.
	Terminal(Stop/Start)	Used to represent start and end of flowchart.
	Input/output	Used for input and output operation.
	Processing	Used for arithmetic operations and data-manipulations.
	Decision	Used to represent the operation in which there are two alternatives, true and false.
	On-page Connector	Used to join different flow line
	Off-page Connector	Used to connect flowchart portion on different page.
	Predefined Process/Function	Used to represent a group of statements performing one processing task.

3. Draw the flowchart to find the maximum number of a given series of data.


Start
Get Array
Set MAX to first number in Array
Set LOOP_INDEX to second Array location
Set END_LOOP to last position of Array location
Is LOOP_INDEX greater than END_LOOP?
Yes
No
loop
MAX contains the largest value in the Array
Is number in Array[LOOP_INDEX] greater than MAX?
Yes
Set MAX to value of number in Array[LOOP_INDEX]
Stop
Increment LOOP_INDEX

4. Explain the following terms of C language with examples.

- **a) Static variables**
- **b) External variables**
- **c) Automatic variables**

a) **Static Variable**:

Variables may be internal or external depending on the place where they are declared. If declared outside main, they are static global. If declared inside main, they are static local. When static, its garbage value is removed and initialized as NULL and then the value remains same throughout the execution.

Eg:

```
#include<stdio.h>
#include<conio.h>
Void main()
{
   int x;
 static int y;
\printf(“x=%d,y=%d”,x,y);
}

Output:
X=1026   //garbage value
  Y=0
```

b)**External Variables(Global Variables):**

Variables which are available to all function are called external variables.They are declared outside the function body. If both global and auto variables have the same name in a program, first priority will be given to auto variables and external variable is hidden.

Eg:

```
   #include<stdio.h>
  #include<conio.h>
void block1();
 void block2();

   void main()
{
      block1();
     block2();
    printf(“n=%d”,n);
}
  void block1()
{
         printf("n=%d",n);
}
  void block2()
{
   printf("n=%d",n);
}
```

Output:

```
  n=10
  n=10
 n=10
```

c) **Automatic Variables:**

Variables inside a function is called automatic variable. The variables by default is auto variable. The scope of the variables are local to the block in which they are defined. Once executed the contents and existence of the automatic variable gets vanished.

Eg:

```
#include<stdio.h>
#include<conio.h>
 void block1();
 void block2();
 void main()
{
  int n=10;
block1();
block2();
printf("n=%d", n);
}
  void block1()
{
    int n=20;
     printf("n=%d",n);
}
  void block2()
{
   int n=30;
   printf("n=%d",n);
}
```

Output:

```
  n=20
  n=30
 n=10
```

5. Write a recursive and non-recursive function or determining the factorial of a given number.

a) using recursive function.

```
#include<stdio.h>
#include<conio.h>
int fact( int);                              // function declaration
int main( )
{
int no,result;
clrscr( );
printf("Enter the required number:");
scanf("%d", &no);
result = fact( no);
printf("\n %d Factorial is : %d", no, result);
getch( );
return 0;
}
int fact(int n)
{
int ft,
if( n==1)
return 1;
else
ft= n*fact (n-1);
return ft;
}
```

b)without using recursive function.

```
#include<stdio.h>
#include<conio.h>
int fact( int);                               // function declaration
int main( )
{
```

```
int no,result;
clrscr( );
printf("Enter the required number:");
scanf("%d", &no);
result = fact( no);
printf("\n %d Factorial is : %d", no, result);
getch( );
return 0;
}
int fact(int n)
{
int ft,
for( ft=1; n>=1; n--)
ft=ft*n;
return ft;
}
```

Output:- 4 Factorial is : 24

Section- C(8 marks Questions)

6. What are the various types of software used in computer? Explain each of them with example.

There are 4 different types of Software used in computer. They are
Application Software
System Software
Programming Software
Driver Software

Application Software

This is the most common type of computer software, and can be defined as end-user programs. Some examples of application software include internet browsers, a CRM tool like Hubspot, a photo-editing software like Adobe or Lightroom, or a word processing application like Microsoft Word. Application software is installed on a computer or mobile device based upon a user's need. Because this is the most common type of software, there are many options available and users can choose the one that best fits their needs, budget, and expectations. (For example, anyone wanting to look on the internet could use Chrome, Safari, or even Firefox.)

System Software

System software helps the user, the computer or mobile device, and an application all work together seamlessly. This makes system software crucial to running any kind of application software as well as the whole computer system. System updating is system software action. Apple’s iOS is an example of system software, as is Microsoft Windows. System software is always running in the background of the device.

Programming-Software

While application software is designed for end-users, and system software is designed for computers or mobile devices, programming software is for computer programmers and developers who are writing code. These are programs that are used to write, develop, test, and debug other software programs. It’s helpful to think of these programs as a translator of sorts: they take programming languages like Laravel, Python, C++, and more and translate them into something a computer or phone will understand.

Driver Software

This software is often considered to be a type of system software. Driver software operates and controls devices that are plugged into a computer. These drivers make it possible for devices to perform their necessary functions. Example: Printer. Setting up printer to work with computer, Driver software has to be installed to connect the two and print data.

7. **What is the specification of a latest PC? Explain the functionality of each of the computer.**

Super-computers are very fast and powerful machines. Their internal architecture enables them to run at the speed of tens of MIPS (Million Instructions per Second). Super-computers are very expensive and for this reason are generally not used for CAD applications. Examples of super-computers are: Cray and CDC Cyber 205.

Mainframe computers are built for general computing, directly serving the needs of business and engineering. Although these computing systems are a step below super-computers, they are still very fast and will process information at about 10 MIPS. Mainframe computing systems are located in a centralized computing center with 20-100+ workstations. This type of computer is still very expensive and is not readily found in architectural/interior design offices.

Minicomputers were developed in the 1960&#39;s resulting from advances in microchip technology.Smaller and less expensive than mainframe computers, minicomputers run at several MIPS and can support 5-20 users. CAD usage throughout the 1960&#39;s used minicomputers due to their low cost and high performance. Examples of minicomputers are: DEC PDP, VAX 11.

Microcomputers were invented in the 1970&#39;s and were generally used for home computing and dedicated data processing workstations. Advances in technology have improved microcomputer capabilities, resulting in the explosive growth of personal computers in industry.

In the 1980&#39;s many medium and small design firms were finally introduced to CAD as a direct result of the low cost and availability of microcomputers. Examples are: IBM, Compaq, Dell, Gateway, and Apple Macintosh. The average computer user today uses a microcomputer. These types of computers include PC&#39;s, laptops, notebooks, and hand-held computers such as Palm Pilots. Larger computers fall into a mini-or mainframe category. A mini-computer is 3-25 times faster than a micro. It is physically larger and has a greater storage capacity.

A mainframe is a larger type of computer and is typically 10-100 times faster than the micro. These computers require a controlled environment both for temperature and humidity. Both the mini and mainframe computers will support more workstations than will a micro. They also cost a great deal more than the micro running into several hundred thousand dollars for the mainframes.

Processors: The term processor is a sub-system of a data processing system which processes received information after it has been encoded into data by the input sub-system. These data are then processed by the processing sub-system before being sent to the output sub-system where they are decoded back into information. However, in common parlance processor is usually referred to the microprocessor, the brains of the modern day computers.

There are two main types of processors: CISC and RISC.

CISC: A Complex Instruction Set Computer (CISC) is a microprocessor Instruction Set Architecture (ISA) in which each instruction can indicate several low-level operations, such as a load from memory, an arithmetic operation, and a memory store, all in a single instruction. The term was coined in contrast to Reduced Instruction Set Computer (RISC).

Examples of CISC processors are the VAX, PDP-11, Motorola 68000 family and the Intel x86/Pentium CPUs.

RISC: Reduced Instruction Set Computing (RISC), is a microprocessor CPU design philosophy that favors a smaller and simpler set of instructions that all take about the same amount of time to execute. Most types of modern microprocessors are RISCs, for instance ARM, DEC Alpha, SPARC, MIPS, and PowerPC.

The microprocessor contains the CPU which is made up of three components--the control unit supervises all that is going on in the computer, the arithmetic/logic unit which performs the math and comparison operation, and temporary memory. Because of the progress in developing better microprocessors, computers are continually evolving into faster and better units.

Notebooks

A laptop computer (also known as notebook computer) is a small mobile personal computer, usually weighing around from 1 to 3 kilograms (2 to 7 pounds). Notebooks smaller than an A4 sheet of paper and weighing around 1 kg are sometimes called sub-notebooks and those weighing around 5 kg a desk note (desktop/notebook). Computers larger than PDAs but smaller than notebooks are also sometimes called &quot;palmtops&quot;. Laptops usually run on batteries.

Notebook Processor:

A notebook processor is a CPU optimized for notebook computers. All computing devices require a CPU. One of the main characteristics differentiating notebook processors from other CPUs is low-power consumption.

The notebook processor is becoming an increasing important market segment in the semiconductor industry. Notebook computers are an increasingly popular format of the broader category of mobile computers. The objective of a notebook computer is to provide the performance and functionality of a desktop computer in a portable size and weight. Wireless networking and low power consumption are primary consideration in the choice of a notebook processor.

Integrated Components

Unlike a desktop computer, a notebook has most of the components built-in or integrated into the computer.

Latest Specifications:

Intel Core i5-8365U Processor.

8GB DDR4 2400 RAM.

120 GB Solid State Drive.

15.6″ Wide Screen Display.

Microsoft Windows Windows 7 Pro(or other least expensive Microsoft Windows operating system)

On board 10/100/1000 Mbps Ethernet NIC.

On board Wireless NIC.

On board HD Audio.

8. **What are the various control structures available in a programming language? Explain with example.**

   The various control structures are Branching and looping statements.

Loops are used in programming to repeat a specific block until some end condition is met. There are three loops in C programming:

1. for loop
2. while loop
3. do...while loop

Example: for loop

```
// Program to calculate the sum of first n natural numbers
// Positive integers 1,2,3...n are known as natural numbers
#include <stdio.h>
int main()
{
   int num, count, sum = 0;
   printf("Enter a positive integer: ");
```

```
    scanf("%d", &num);
    // for loop terminates when n is less than count
    for(count = 1; count <= num; ++count)
    {
        sum += count;
    }
    printf("Sum = %d", sum);
    return 0;
}
```

```
Output    :    Enter a positive integer: 10                    Sum = 55
```

The while loop evaluates the test expression.

If the test expression is true (nonzero), codes inside the body of while loop are exectued. The test expression is evaluated again. The process goes on until the test expression is false.

When the test expression is false, the while loop is terminated.

```
// Program to find factorial of a number
// For a positive integer n, factorial = 1*2*3...n
#include <stdio.h>
int main()
{
    int number;
    long long factorial;
    printf("Enter an integer: ");
    scanf("%d",&number);
    factorial = 1;
    // loop terminates when number is less than or equal to 0
    while (number > 0)
    {
        factorial *= number;  // factorial = factorial*number;
        --number;
    }
    printf("Factorial= %lld", factorial);
 return 0;
}
```

```
Output        Enter an integer: 5                    Factorial = 120
```

do...while loop

The do..while loop is similar to the while loop with one important difference. The body of do...while loop is executed once, before checking the test expression. Hence, the do...while loop is executed at least once.

do...while loop Syntax

```
do
{
  // codes
}
while (testExpression);
```

```
// Program to add numbers until user enters zero
#include <stdio.h>
int main()
{
   double number, sum = 0;
   // loop body is executed at least once
   do
   {
      printf("Enter a number: ");
      scanf("%lf", &number);
      sum += number;
   }
   while(number != 0.0);
   printf("Sum = %.2lf",sum);
   return 0;
}
```

Output

```
Enter a number: 1.5          Enter a number: 2.4          Enter a number: -3.4
Enter a number: 4.2          Enter a number: 0                   Sum = 4.70
```

Difference between while and do-while loop.

C Branching Statements:

if statement

The if statement is a powerful decision making statement which can handle a single condition or group of statements. These have either true or false action....

```
#include< stdio.h >
#include< conio.h >
void main()
{
   int n;
   n=1;
   clrscr(); //clrscr() is the function of #include< conio.h > header file which will clear previous output of program
   printf("Enter the Number");
   scanf("%d",&n);
   if(n>0)
   {
      printf("It is If Statement");
   }
   getch();
}
```

if-else statement

```
#include< stdio.h >
#include< conio.h >
void main()
{
   int n;
   n=1;
   clrscr();
   printf("Enter the Number");
   scanf("%d",&n);
   if(n%2==0)
   {
      printf("This is Even Number");
   }
   else
   {
      printf("This is Odd Number");
   }
   getch();
}
```

Output is as :

```
Enter the Number
4
This is Even Number
```

Nested if statement

When an if statement occurs within another if statement, then such type of is called nested if statement.

/*If the ages of Ram, sham and Ajay are input through the keyboard, write a program to determine the youngest of the three*/

```
#include< stdio.h >
#include< conio.h >
void main()
{
  int ram,sham,ajay;
  clrscr();
  printf("Enter the Three Ages of Ram,Sham and Ajay\n");
  scanf("%d%d%d",&ram,&sham,&ajay);
  if(ram < sham)
  {
    if(ram < ajay)
    {
      printf("Ram is Youngest");
    }
    else
    {
      printf("Ajay is Youngest");
    }
  }
  else
  {
    if(sham < ajay)
    {
      printf("Sham is Youngest");
    }
    else
    {
      printf("Ajay is Youngest");
    }
  }
  getch();
}
```

```
Output is as :
Enter the three Ages of Ram,Sham and Ajay
14
17
19
Ram is Youngest
```

Switch statement

When number of conditions (multiple conditions) occurs in a problem and it is very difficult to solve such type of complex problem with the help of ladder if statement, then there is need of such type of statement which should have different alternatives or different cases to solve the problem in simple and easy way. For this purpose switch statement is used. .

/*WAP to print the four-days of week from monday to thrusday which works upon the choice as S,M,T,H using switch case*/

```
#include< stdio.h >
#include< conio.h >
void main()
{
  char n;
  clrscr();
  printf("Enter the Choice from Four Days...\n")
  printf("S = Sunday \n")
  printf("M = Monday \n")
  printf("T = Tuesday \n")
  printf("H = Thursday \n\n")
  scanf("%c",&n);
  switch(n)
  {
     case 'S':
     printf("Sunday");
     break;
     case 'M':
     printf("Monday");
     break;
     case 'T':
     printf("Tuesday");
     break;
```

```
      case 'H':
      printf("Thursday");
      break;
      default:
      printf("Out of Choice");
      break;
    }
  getch();
}
```

Output:

```
Enter the Choice from Four Days...
S = Sunday    , M = Monday    , T = Tuesday  , H = Thursday ,  S= Sunday
```

9. Write a program to find difference of two matrix using function.

```
#include <stdio.h>
    #include <stdlib.h>
     void readmatA();
    void printmatA();
    void readmatB();
    void printmatB();
    void diff();
     int a[10][10], b[10][10], arraydiff[10][10];
    int i, j, row1, column1, row2, column2;
     void main()
    {
         printf("Enter the order of the matrix A \n");
        scanf("%d %d", &row1, &column1);
         printf("Enter the order of the matrix B \n");
        scanf("%d %d", &row2, &column2);
         if (row1 != row2 && column1 != column2)
        {
            printf("Addition and subtraction are possible \n");
            exit(1);
        }
         else
        {
             printf("Enter the elements of matrix A \n");
            readmatA();
```

```
       printf("MATRIX A is \n");
      printmatA();
       printf("Enter the elements of matrix B \n");
      readmatB();
       printf("MATRIX B is \n");
      printmatB();
       diff();
    }
 }
 /*  Function to read a matrix A */
 void readmatA()
{
   for (i = 0; i < row1; i++)
   {
      for (j = 0; j < column1; j++)
      {
         scanf("%d", &a[i][j]);
      }
   }
   return;
}
 /*  Function to read a matrix B */
 void readmatB()
{
   for (i = 0; i < row2; i++)
   {
      for (j = 0; j < column2; j++)
      {
         scanf("%d", &b[i][j]);
      }
   }
}
 /*  Function to print a matrix A */
 void printmatA()
{
   for (i = 0; i < row1; i++)
   {
      for (j = 0; j < column1; j++)
      {
            printf("%3d", a[i][j]);
         }
         printf("\n");
      }
   }
   /*  Function to print a matrix B */
   void printmatB()
  {
      for (i = 0; i < row2; i++)
      {
         for (j = 0; j < column2; j++)
         {
            printf("%3d", b[i][j]);
         }
         printf("\n");
      }
   }
      /*  Function to do the difference of elements of matrix A and Matrix B */
   void diff()
  {
      for (i = 0; i < row1; i++)
      {
         for (j = 0; j < column2; j++)
         {
            arraydiff[i][j] = a[i][j] - b[i][j];
         }
       }

      printf("Difference matrix is \n");
      for (i = 0; i < row1; i++)
      {
         for (j = 0; j < column2; j++)
         {
            printf("%3d", arraydiff[i][j]);
          }
         printf("\n");

      }
      return;
```

```
    }
```

Output:

```
Enter the order of the matrix A
3 3
Enter the order of the matrix B
3 3
Enter the elements of matrix A
1 4 5
6 7 8
4 8 9
MATRIX A is
  1  4  5
  6  7  8
  4  8  9
Enter the elements of matrix B
3 6 7
8 4 2
1 5 3
MATRIX B is
  3  6  7
  8  4  2
  1  5  3
Difference matrix is
 -2 -2 -2
 -2  3  6
  3  3  6
```