CH # 4 DATA AND REPETITION

4.1 DATA STRUCTURE

SHORT QUESTIONS

Q.1 Define Data Structure. (K.B)

Ans: Data structure is a container to store collection of data items in a specific layout.

Q.2 How many forms do data structure have? (K.B)

Ans: There are generally four forms of data Structures:

- Linear: arrays, lists.

MULTIPLE CHOICE QUESTIONS

1. _____ structure is a container to store data items. (K.B)

(A) Data (B) Control (C) Selection (D) Loop

2. Data structure is a container to store data in _____ layout. (K.B)

(A) Specific (B) Irregular (C) Alternative (D) None of these

3. _____ is type of data structure: (K.B)

(A) Array (B) List (C) Both A & B (D) None of these

4.1.1 ARRAY
4.1.2 ARRAY DECORATION
4.1.3 ARRAY INITIALIZATION
4.1.4 ACCESSING ARRAY ELEMENTS
4.1.5 USING VARIABLES AS ARRAY INDEXES.

LONG QUESTION

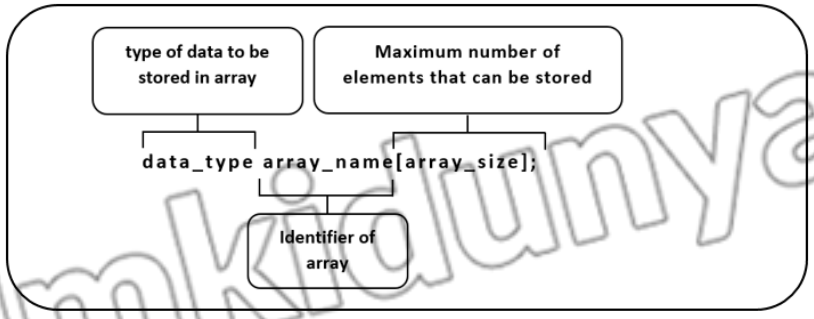
1. Briefly Explain Array in detail. (K.B+U.B)

Ans: **Definition**

An array is a data structure that can hold multiple values of same data type e.g. an 'int' array can hold multiple integer values, a 'float' array can hold multiple real values and so on. An important property of array is that it stores all the values at consecutive locations inside the computer memory.

Array Declaration:

In C language, an array can be declared as follows:



Array Initialization:

Assigning values to an array for the first time, is called array initialization. An array can be initialized at the time of its declaration, or later. Array initialization at the time of declaration can be done in the following manner.

Data_type array_name[N] = {value1, value2, value3,……, value N};

Char vowels [5] = {'a', 'e', 'i', 'o', 'u'}

If we do not initialize an array at the time of declaration. Then we need to initialize the array elements one by one. It means that we cannot initialize all the elements of array in a single statement.

Elements of Array:

Each element of an array has an index that can be used with the array name as array-name [index] to access the data stored at that particular index.
First element has the index 0, second element has the index 1 and so on. Thus height [0] refers to the first element of array height, height [1] refers to the second element and so on. Figure shows graphical representation of array height initialized in the last section.

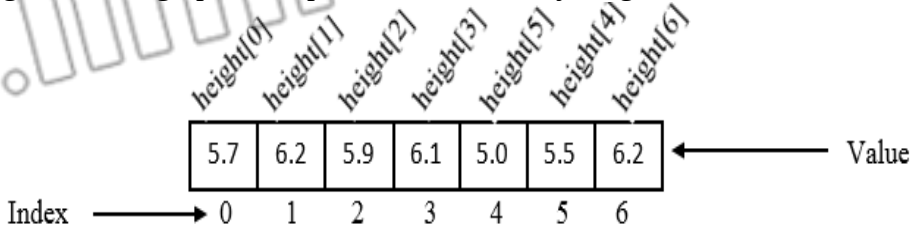


Figure 4.1: Graphical representation of array height

Example:
Write a program that stores the ages of five persons in an array, and then displays on screen.
Solution:

```
#include<stdio.h>
void main()
{
int age [5];
/* Following statements assign values at different indices of array age, we can see that the first value is stored at index 0 and the last value is stored at index 4 */
age [0] = 25;
age [1] = 34;
age [2] = 29;
age [3] = 43;
age [4] = 19;
/* Following statement displays the ages of five persons stored in the array *\
printf("The ages of five persons are: %d, %d, %d, %d, %d", age [0], age [1], age [2], age [3], age [4]);
```

SHORT QUESTIONS

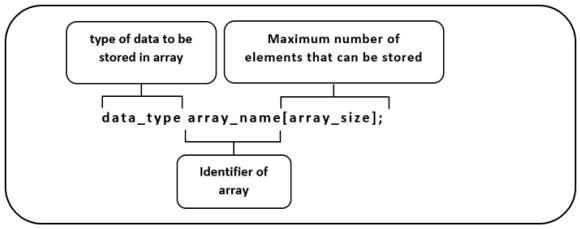
Q.1 Define Array. (K.B)

Ans: An array is a data structure that can hold multiple values of same data type e.g. an 'int' array can hold multiple integer values, a 'float' array can hold multiple real values and so on. An important property of array is that it stores all the values at consecutive locations inside the computer memory.

Data_type array_name[N] = {value1, value2, value3,……, value N};

Q.2 Draw a diagram to show array declaration. (K.B+U.B+A.B)

Ans:



Q.3 Write down the syntax to initialize an Array. (U.B+A.B)

Ans: *Data_type array_name[N] = {value1, value2, value3,……, value N};*

Q.4 Define Array Initialization. (K.B)

Ans: Assigning values to an array for the first time, is called array initialization. An array can be initialized at the time of its declaration, or later. Array initialization at the time of declaration can be done in the following manner.

Data_type array_name[N] = {value1, value2, value3,……, value N};

Q.5 Define Element of an Array. (K.B)

Ans: Each element of an array has an index that can be used with the array name as array-name[index] to access the data stored at that particular index.

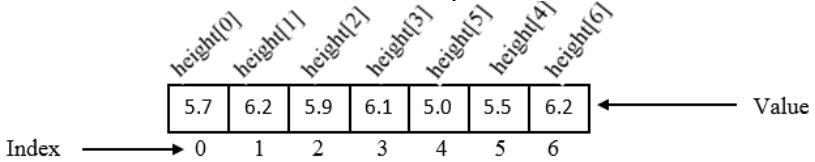


Figure 4.1: Graphical representation of array height

MULTIPLE CHOICE QUESTIONS

1. Each element of array has _____. (K.B)

(A) Value (B) Index (C) Subscript (D) Both B & C

2. _____ is the by default index. (K.B)

(A) 0 (B) 1 (C) 2 (D) Any

3. _____ can be used as any indexes. (K.B+U.B)

(A) Variable (B) Constant (C) Both A & B (D) None of these

4. Index can be _____.

(A) Variable (B) Constant (C) Both A & B (D) None of these

4.2 LOOP STRUCTURE
4.2.2 FOR LOOP
4.2.3 NESTED LOOP
4.2.5 LOOPS AND ARRAYS

LONG QUESTION

1. Define Loop. Explain *for* Loop in detail. (K.B+U.B)

Ans: Definition

"Loop is a control structure that repeats a statement or set of statements up to a fix number of times or until a specific condition is satisfied**"**

Types of loops:

'C' language provides three kind of loop structure:

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

***for* Loop:**

Definition:

"*for"* loop is a type of loop which keeps on repeating a statement or a set of statements up to a fixed number of times**"**

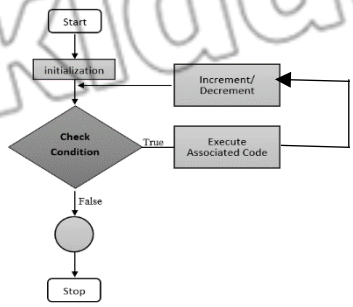
General Syntax:

In C programming language, ***for*** loop has the following general syntax.

```
for(variable_initialization; condition; increment/decrement)
{
Code to repeat
}
```

Flowchart:

In order to understand the ***for*** loop structure let's look at the following flow chart.



From the flow chart, we can observe the following sequence:

Step 1. Initialization is the first part to be executed in a ***for*** loop. Here we initialize our counter variable and then move to the condition part.

Step 2. Condition is checked, and if it turns out to be false, then we come out of loop.

Step 3. If the condition is true, then body of the loop is executed.

Step 4. After executing the body of loop, the counter variable is increased or decreased depending on the used logic and then we move again to the **step 2**.

Example:

```
for(int i = 1; i < = 10; i++)
{
printf("%d\n", i);
}
```

Output:

1
2
3
4
5
6
7
8
9
10

4.2.5 LOOPS AND ARRAYS

LONG QUESTIONS

Q.1 Relate Loops with Arrays. Also discuss how loop can be used to read and write values in arrays?

As variables can be used as array indexes, so we can use loops to perform different operations on array. If we want to display the whole array, then instead of writing all the elements on by one, we can loop over the array elements by using the loop counter as array index.

In the following, we discuss how loops can be used to read and write values in arrays.

1. Writing values in Array using Loops:

Using loops, we can easily take input in arrays. If we want to take input from user in an array of size 10, we can simply use a loop as follows:

Example:

```
int a [10];
for (int i =0, i < 10; i++)
scanf("%d", &a[i]);
```

2. Reading values from Arrays using Loops:

Loops help us in reading the values from array. The following code can be used to display the elements for an array having 100 elements:

Example

```
for (int i = 0; i < 100; i++)
printf("%d", a[i]);
```

SHORT QUESTIONS

Q.1 Define Loop. (K.B)

Ans: **"**Loop is a control structure that repeats a statement or set of statements up to a fix number of times of until a specific condition is satisfied**"**

Q.2 Write the names of different types of loops. (K.B)

Ans: C language provides three kind of loop structure:

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

Q.3 Define *'for'* Loop. (K.B)

Ans: **"*for*"** loop is a type of loop which keeps on repeating a statement or a set of statements up to a fixed number of times**"**

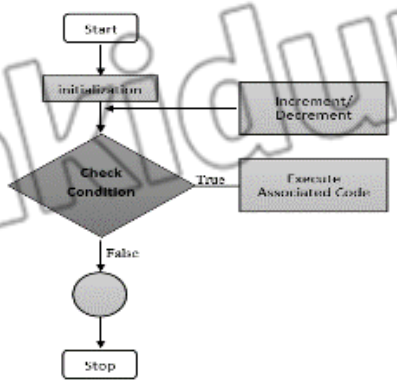
Q.4 Write down the syntax of '*for'* loop. (K.B+U.B)

Ans: In C programming language, '*for'* loop has the following general syntax.

```
for (initialization; condition; increment/decrement)
{
Code to repeat
}
```

Q.5 Draw flowchart to explain '*for'* loop. (K.B+U.B+A.B)

Ans:



Q.6 Write a program to print "Pakistan" three times using '*for'* loop. (A.B)

Ans:

```
for(int i = 0; i < 3; i++)
{
```

```
    printf("Pakistan\n");
}
```

Q.7 Write a program to display the values from 1 to 10 on screen. (A.B)

Ans:

```
for (int i = 1; i < = 10; i++)
{
    printf("%d\n", i);
}
```

Q.8 Write a program that displays the table of 2. (A.B)

Ans:

```
#include <stdio.h>
void main()
{
int n=2,j;
for(j=1;j<=10;j++)
{
      printf("%d X %d = %d \n",n,j,n*j);
}
}
```

Q.9 What will happen if condition don't return *false* in any case of '*for*' loop? (K.B+U.B)

Ans: If the condition in loop do not gets false at some point then the loop repeats for infinity and never terminates

Q.10 Define iteration. (K.B)

Ans: Each run of loop is called an iteration

Q.11 Define Nested '*for*' Loop. (K.B)

Ans: A loop in a loop is known as nested loop.

Q.12 Write down the structure of Nested '*for*' loop. (K.B)

Ans:

```
for (initialization; condition; increment/decrement)
{
for (initialization; condition; increment/decrement)
{
    Code to repeat
}
}
```

Q.13 Write a program that shows the working of nested '*for*' loop. (A.B)

Ans:

```
#include<stdio.h>
void main()
for(int i =1; i < = 5; i++)
{
for(int j =1; j < = i; j++)
{
printf("*");
}
printf("\n");
```

```
}
}
```

Q.14 Write a program that displays the table of 2, 3, 4, 5 and 6. (A.B)

Ans:

```
#include <stdio.h>
        void main()
{
        int n=2,j;
        printf("TABLE OF 2");
        for(j=1;j<=10;j++)
  {
        printf("%d x %d = %d \n",n,j,n*j);
  }
        n=3;
        printf("\nTABLE OF 3");
        for(j=1;j<=10;j++)
  {
        printf("%d x %d = %d \n",n,j,n*j);
  }
        n=4;
        printf("\nTABLE OF 4");
        for(j=1;j<=10;j++)
  {
        printf("%d x %d = %d \n",n,j,n*j);
  }
        n=5;
        printf("\nTABLE OF 5");
        for(j=1;j<=10;j++)
  {
        printf("%d x %d = %d \n",n,j,n*j);
  }
        n=6;
        printf("\nTABLE OF 6");
        for(j=1;j<=10;j++)
  {
        printf("%d x %d = %d \n",n,j,n*j);
  }
}
```

Q.15 How loops can be used to read and write values in arrays. (K.B+U.B)

Ans: Writing values in Array using Loops:

Using loops, we can easily take input in arrays. If we want to take input from user in an array of size 10, we can simply use a loop as follows:

```
int a [10];
```

```
for (int i =0, i < 10; i++)
scanf("%d", &a[i]);
```

Reading values from Arrays using Loops: Loops help us in reading the values from array. The following code can be used to display the elements for an array having 100elements:

```
for (int i = 0; i < 100; i++)
printf("%d", a[i]);
```

MULTIPLE CHOICE QUESTIONS

1. A structure that enables the programmer to execute same sequence of statement repeatedly until a particular condition is met is called _____. (K.B)

(A) Loop (B) Sequence (C) Selection (D) Switch Statement

2. A loop has essential elements _____. (K.B+U.B)

(A) Two (B) Three (C) Four (D) Five

3. A loop terminates based on _____. (K.B+U.B)

(A) Test Condition (B) Block of Statement
(C) Increment (D) Decrement

4. A statement that is executed fixed number of times is called _____. (K.B)

(A) for Statement (B) while Statement
(C) do–while Statement (D) Nested loop Statement

5. A loop that is known as counter loop _____. (K.B)

(A) For (B) While (C) Do – While (D) Nested

6. When a *for Loop* is executed a variable is assigned on initial value in the loops called _____. (K.B+U.B)

(A) Loop variable (B) Variable (C) Increment (D) Decrement

7. The general syntax of *for* loop is _____. (K.B+U.B)

(A) for (initialization; test condition; increment / decrement)
(B) for (test condition; increment)
(C) for (initialization; test condition; post fix)
(D) for (variable; condition; and decrement)

8. What will be the output of following C code? (K.B+U.B+A.B)

```
#include <stdio.h>
void main()
{
int k =0;
for (k)
printf("Hello");
}
```

(A) Compile Time Error (B) Hello
(C) Nothing (D) Varies

9. What will be the output of following C code? (K.B+U.B+A.B)

```
#include <stdio.h>
void main()
{
int k;
for (k = -3 ; k < -5; k++)
printf("Hello");
}
```

(A) Hello (B) Infinite Hello (C) Run Time Error (D) Nothing

10. What will be the output of following C code? (K.B+U.B+A.B)

```
for (i=0; i<10; i++)
```

printf("%d", i);

(A) 10 (B) 0123456789 (C) 0 (D) Syntax Error

11. What will be the output of following C code? (K.B+U.B+A.B)

for(i=0; i++; i<15)

printf("%d ", i);

(A) 10 11 12 13 14 (B) 9 10 11 12 13

(C) Infinite Loop (D) 10 11 12 13 14 15 16

12. What will be the final value of the variable 'digit'? (U.B+A.B)

```
void main()
{
int digit = 0;
for( ; digit < = 9; )
digit++;
digit *= 2;
digit;
}
```

(A) -1 (B) 17 (C) 19 (D) 26

13. Which one of the following is not a loop statement? (K.B)

(A) for (B) if (C) while (D) do – while

14. When number of iterations of execution are known in advance, which is the best choice of programmer? (K.B+U.B)

(A) for (B) do – while (C) while (D) None of These

15. How many expressions are used in *for* Loop? (K.B)

(A) 2 (B) 4 (C) 5 (D) 3

16. Which are the mandatory part of *for* Loop? (K.B+U.B)

(A) Initialization (B) Condition

(C) Increment/Decrement (D) All of these

17. The *for* loop expressions are enclosed in: (K.B+U.B)

(A) ( ) (B) { } (C) [ ] (D) None of These

18. Which of the following expression is executed only once in the *for* Loop? (K.B+U.B)

(A) Condition Loop Control (B) Increment/Decrement
(C) Initialization (D) None of These

19. *for* loop is also called: (K.B)

(A) Conditional Loop (B) Counter Loop (C) Sentinel Loop (D) Nested Loop

20. The expressions in the *for* loop is separated by: (K.B+U.B)

(A) , (B) ; (C) ¦ (D) .

21. Which part of loop statement is used to control the loop? (K.B+U.B)

(A) Body (B) Increment /Decrement

(C) Condition (D) None of These

22. The loop which never end is called: (K.B+U.B)

(A) Running Loop (B) Infinite Loop (C) Nested Loop (D) Continuous Loop

23. Which one of the following is loop statement? (K.B+U.B)

(A) if (B) if-else (C) switch (D) None of These

24. One execution of a loop is known as: (K.B)

(A) Cycle (B) Duration (C) Iteration (D) Both A & C

25. In *for* loop, this expression is executed only once: (K.B+U.B)

(A) Test (B) Validation (C) Initialization (D) None of these

26. -------- is known as repetition structure. (K.B+U.B)

(A) Loop (B) Selection (C) Sequential (D) None of these

27. --------- repeats one or more statement. (K.B+U.B)

(A) Loop (B) Selection (C) Sequential (D) None of these

28. There are ---- types of loop. (K.B)

(A) 1 (B) 2 (C) 3 (D) 4

29. -------- loop is used to repeat a statement to a fix number of times. (K.B+U.B)

(A) for (B) Do (C) do-white (D) None of these

30. ------- is a first part to be executed in *for* loop. (K.B+U.B)

(A) Initialization (B) condition (C) increment (D) None of these

31. -------- is represented by 1

(A) True (B) False (C) Both A & B (D) None of these

32. Loop in a loop is called _____. (K.B+U.B)

(A) nested loop (B) for loop (C) do loop (D) None of these

33. Each run of a loop is called _____. (K.B)

(A) cycle (B) iteration (C) Both A & B (D) None of these

PROGRAMMING TIME (A.B

Programming Time 4.1

Write a program that stores the ages of five persons in an array, and then displays on screen.

Solution:

```
#include<stdio.h>
void main ()
{
        int age[5];
        /* Following statements assign values at different indices of array age.We can
        see that the first value is stored at index 0 and the last value is stored at index
        4 */
                age[0]   = 25;
                age[1]   = 34;
                age[2]   = 29;
                age[3]   = 43;
                age[4]   = 19;
/* Following statement displays the ages of five persons stored in the array  * /
.
printf("The ages of five persons are: %d, %d, %d, %d, %d", age[0], age[1], age[2],
                age[3], age[4]);
```

Programming Time 4.2

Write a program that takes the marks obtained in 4 subjects as input from the user, calculates the total marks and displays on screen.

Solution:

```
#include<stdio.h>
void main()
{
        float marks[4], total_marks;
        printf("Please enter the marks obtained in 4 subjects:")
        scanf("%f%f%f%f, &marks[0], &marks[1], &marks[2],&marks[3]);
        total_marks  = marks[0]+marks[1]+marks[2]+marks[3];
        printf("Total  marks  obtained  by· student  are  %f",total_marks);
}
```

Programming Time 4.3

Write a program that displays the values from 1 -- 10 on the computer screen.

Program:

```
for(int i=1;i<=10;i++)
{
        printf("%d\n",i);
```

}
Description:
Consider the example program given above.

- First of all, the value of i is set to 1 and then condition is checked.
- As the condition is true (1<= 10) so loop body executes. As in the loop body, we are displaying the value of the counter variable, so 1 is displayed on console.
- After increment, the value of i becomes 2. The condition is again checked. It is true as(2<=10) so this time 2 is printed.
- The procedure continues till 10 is displayed and after increment the value of i becomes 11. Condition is checked and it turns out to be false (11>10) so the loop finally terminates after printing the numbers from 1 to 10.

Programming Time 4.4

Write a program that calculates the factorial of a number input by user.
Program Logic:
When we want to solve a problem programmatically, first we need to know exactly what we want to achieve. In this example, we are required to find the factorial of a given number, so first we need to know the formula to find factorial of a number.

N! = 1 * 2 * 3 * 4 * ..... * (N-1) * N

We can see the pattern that is being repeated, so we can solve the problem using for loop.
Program:

```
#include<stdio.h>
void main()
{
        int n, fact = 1;
        printf("Please enter a positive number whose factorial you want to find");
        scanf("%d, &n);
        for(int I = 1; i <=n; i++)
        {
                fact = fact * i;
        }
        printf("The factorial of input number %d is %d, n, fact);
}
```

Description:
Following table shows the working of program, if the input number is 5. It demonstrates the changes in the values of variables at each iteration.

Iteration	Value of counter	Condition	Loop body	Result
				fact=1
1	i=1	TRUE(1<=5)	fact = fact*i	fact=1*1=1.
2	i=2	TRUE(2<=5)	fact = fact*i	fact=1*2=2
3	i=3	TRUE(3<=5)	fact = fact*i	fact=2*3=6

4	i=4	TRUE(4<=5)	fact = fact*i	fact=6*4=24
5	i=5	TRUE(5<=5)	fact = fact*i	fact=24*5=120
6	i=6	FALSE(6>5)		

Programming Time 4.5

Problem:
Write a program that 5 times displays the numbers from 1-10 on computer screen.
Program:

```
#include<stdio.h>
void main()
{
        for(int i = 1; i <= 5; i++)
        {
                for(int j = 1; j <=10; j++)
                {
                        printf(%d", j);
                }
                printf("\n");
        }
```

}

Output:

Here is the output of above program.

```
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10
```

Description:

As we understand the working of inner loop, so here let's focus on outer loop.

1- For the value i = 1, condition in outer loop is checked which is true (1 < = 5), so whole inner loop is executed and numbers from 1 – 10 are displayed.
2- When control gets out of inner loop, printf("\n"); is executed which inserts a new line on console.
3- Then i is incremented and it becomes 2. As it is less than 5, so condition is true. The whole inner loop is executed, and thus numbers from 1 – 10 are again displayed on screen. Coming out of the inner loop new line is inserted again.
4- After five times displaying the numbers from 1 – 10 on screen, the value of i gets incremented to 6 and condition of outer loop turns false. So outer loop also terminates.

Programming Time 4.6

Problem:

Write a program to display the following pattern of stars on screen.

```
*
**
***
****
*****
******
```

Program:

```
#include<stdio.h>
void main ( )
{
        for(int i = 1; i <= 6; i++)
        {
                for(int j = 1; j <= i; j++)
                        printf("*");
                printf("\n");
        }
}
```

Description:

Here is the description of above code.

1- As we have to display 6 lines containing stars, so we run the outer loop from 1 to 6.
2- We can observe that in the given pattern we have 1 star on 1st line, 2 stars on 2nd line, 3 stars on 3rd line and so on. So, the inner loop is dependent on the outer loop, i.e. if counter of outer loop is 1 then inner loop should run 1 time,

if the counter of outer loop is 2 then inner loop should run 2 times and so on. So, we use the counter of outer loop in the termination condition of inner loop i.e. j <= i.

3- When outer loop counter i has value 1, inner loop only runs 1 time, so only 1 star is displayed. When outer loop counter is 2, the inner loop runs 2 times, so 2 stars are displayed and the process is repeated until six lines are complete.

Programming Time 4.7

Problem:

Write a program that counts multiples of a given number lying between two numbers.

Program:

```
#include <stdio.h>
void main ()
{
        int n, lower, upper, count = 0;
        printf (“Enter the number:  ”);
        scanf (“%d", &n);
        printf (“Enter the lower and upper limit of multiples:\n");
        scanf (“%d%d”, &lower, &upper);
        for(int i = lower; i <= upper; i++)
                if(i%n == 0)
                        count++;
        printf ("Number of multiples of %d between %d and %d are %d”, n, lower,
        upper, count);
}
```

Programming Time 4.8

Problem:

Write a program to find even numbers in integers ranging from n1 to n2 (where nl is greater than n2).

Program:

```
#include  <stdio.h>
void main ( )
{
        int n1, n2;
        printf (“Enter the lower and upper limit of even numbers: \n”)
        scanf (“%d%d", &n2, &n1);
        if(n1 > n2)
        {
                for (int i = n1; i > = n2; i--)
                {
```

```
                    if(i % 2 == 0)
                        printf ("%d", i);
                }
        }
}
```

Programming Time 4.9

Problem:

Write a program to determine whether a given number is prime number or not.

Program:

```
#include <stdio.h>
void main ( )
{
        int n;
        int flag = 1;
        printf ("Enter a number:  ");
        scanf ("%d", &n);
        for (int i = 2; i < n; i++)
        {
                if (n % i == 0)
                        flag = 0;
        }
        if (flag == 1)
                printf ("This is a prime number");
        else
                printf ("This is not a prime number");
}
```

Programming Time 4.10

Problem:

Write a program to display prime numbers ranging from 2 to 100.

Program:

```
# include<stdio.h>
int main ()
{
        int flag;
        for (int j = 2; j <= 100;  j++)
        {
                flag = 1;
```

```
                for (int i = 2; i < j; i++)
                {
                        if(j % i == 0)
                        {
                                flag = 0;
                        }
                }
                if (flag == 1)
                {
                        printf ("%d", j);
                }
        }
}
```

Programming Time 4.11

Problem:
Write a program that assigns first 5 multiples of 23 to an array of size 5.
Program:

```
#include<stdio.h>
void main( )
{
        int multiples [5];
        for (int i = 0; i < 5; i++)
                multiples [i] = (i + 1) * 23;
}
```

Programming Time 4.12

Problem:
Write a program that adds corresponding elements of two arrays.
Program:

```
#include <stdio.h>
void main ( )
{
        int a[ ] = {2, 3, 54, 22, 67, 34, 29, 19);
        int b[ ] = {65,73, 26, 10, 4, 2, 84, 26);
        for (int i=0; i<8; i++)
                printf ("%d ", a[i] +b[i]);
}
```

SOLVED ACTIVITIES (A.B

ACTIVITY 4.1

Write a program that displays the table of '2'.
Solution:

```
#include <stdio.h>
void main()
{
int n=2,j;
for(j=1;j<=10;j++)
```

```
{
printf("%d X %d = %d \n",n,j,n*j);
}
}
```

ACTIVITY 4.2

Write a program that displays the table of 2,3,4,5 and 6.

Solution:

```
#include <stdio.h>
void main()
{
int n=2,j;
for(j=1;j<=10;j++)
{
printf("%d x %d = %d \n",n,j,n*j);
}
n=3;
for(j=1;j<=10;j++)
{
printf("%d x %d = %d \n",n,j,n*j);
}
n=4;
for(j=1;j<=10;j++)
{
printf("%d x %d = %d \n",n,j,n*j);
}
n=5;
for(j=1;j<=10;j++)
{
printf("%d x %d = %d \n",n,j,n*j);
}
n=6;
for(j=1;j<=10;j++)
{
printf("%d x %d = %d \n",n,j,n*j);
}
}
```

ACTIVITY 4.3

Write a program that takes as input the marks obtained in matriculation by 30 students of a class. The program should display the average marks of the class.

Solution:

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

```
int i;
int Avg;
int maks[30];                          //Array Declaration
int sum=0;
for( i=0 ; i<=29 ; i++)
{
printf(" Enter Marks/n ");
scanf("%d" , &marks[i] );              // Store Data in      Array
}
for( i=0 ; i<=29 ; i++ )
{
Sum = sum + marks[i];                  // Read Data  from an Array
Avg = Sum/30;
printf(" Average Marks = %d\n" , Avg);
}
}
```


EXERCISE

Q1. Multiple Choice Questions

1) An array is a_____structure. **(K.B)**
A) Loop B) Control C) Data D) Conditional

2) Array elements are stored at _____ memory locations. **(K.B)**
A) Contiguous B) Scattered C) Divided D) None

3) If the size of an array is 100, the range of indexes will be _____. **(K.B)**
A) 0-99 B) 0-100 C) 1-100 D) 2-102

4) _____ structure allows repetition of a set of instructions. **(K.B+U.B)**
A) Loop B) Conditional C) Control D) Data

5) _____ is the unique identifier, used to refer to the array. **(K.B)**
A) Data Type B) Array Name C) Array Size D) None

6) Array can be initialized _____ declaration. **(K.B)**

A) At the time of B) After C) Before D) Both a & b

7) Using loops inside loops is called _____loops. (K.B+U.B)

A) For B) While C) Do-while D) Nested

8) _____ part of for loop is executed first. (K.B)

A) Condition B) Body

C) Initialization D) Increment/Decrement

9) _____ make it easier to read and write values in array. (K.B+U.B)

A) Loops B) Conditions C) Expressions D) Functions

10) To initialize the array in a single statement, initialize it_ declaration. (K.B+U.B)

A) At the time of B) After C) Before D) Both A & B

ANSWER KEY

1	2	3	4	5	6	7	8	9	10
C	A	A	A	B	D	D	C	A	A

Q2. Define the following terms. (K.B)

1) Data Structure

Ans: Data structure is a container to store collection of data items in a specific layout. Array is the most commonly used Data Structure of C language.

2) Array

Ans: An array is a data structure that can hold multiple values of same data type e.g. an int array can hold multiple integer values, a float array can hold multiple real values and so on. An important property of array is that it stores all the values at consecutive locations inside the computer memory.

Syntax

Data_type array name [N] = {value1, value2, value3,……, value N};

3) Array Initialization

Ans: Assigning values to an array for the first time, is called array initialization. An array can be initialized at the time of its declaration, or later. Array initialization at the time of declaration can be done in the following manner.

Data_type array_name[N] = {value1, value2, value3,……, value N};

4) Loop Structure

Ans: If we need to repeat one or more statements, then we use loops. For example, if we need to write Pakistan thousand times on the screen then instead of writing printf("Pakistan"); a thousand times, we use loops. C language provides three kind of loop structure:

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

5) Nested Loops

Ans: A loop within a loop is known as nested loop.

General Structure

```
for(initialization; condition; increment/decrement)
{
for(initialization; condition; increment/decrement)
{
Code to repeat
}
}
```

Q3. Briefly answer the following questions. (K.B+U.B)

1) Is loop a data structure? Justify your answer.

Ans: No Loop is not a data structure. Loop is a type of control structure which repeats a statement of set of statements. While Data Structure is a container to store collection of data items in a specific layout.

2) What is the use of nested loops?

Ans: When we use a loop inside another loop, it is called nested loop structure. We use nested loops to repeat a pattern multiple time.

General Structure

```
for (initialization; condition; increment/decrement)
{
for (initialization; condition; increment/decrement)
{
Code to repeat
}
}
```

3) What is the advantage of initializing an array at the time of declaration?

Ans: If we do not initialize an array at the time of declaration. Then we need to initialize the array elements one by one. It means that we cannot initialize all the elements of array in a single statement. This is demonstrated by the following example.

Example:

```
void main()
{
        Int array [5];
        Array[5]= {10, 20, 30, 40, 50};
}
```

ERROR Initializing whole array after declaration not allowed

The compiler generates an error on the above example code, as we try to initialize the whole array in one separate statement after declaring it.

4) Describe the structure of a for loop.

Ans:
```
for (initialization; condition; increment/decrement)
{
Code to repeat
}
```

Step 1. Initialization is the first part to be executed in a for loop. Here we initialize our counter variable and then move to the condition part.

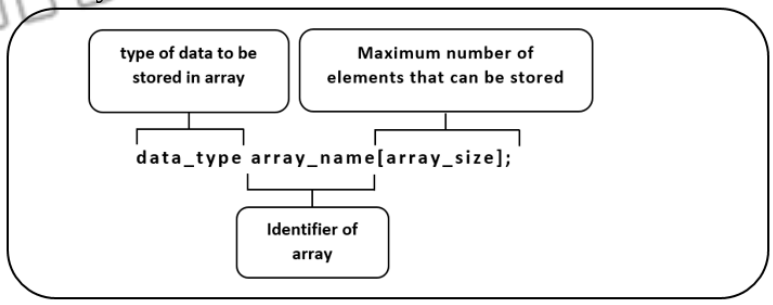
Step 2. Condition is declared and checked, and if it turns out to be false, then we come out of loop.

Step 3. If the condition is true, then body of the loop is executed.

Step 4. After executing the body of loop, the counter variable is increased or decreased depending on the used logic, and then we again move to the **step 2**.

5) How can you declare an array? Briefly describe the three parts of array declaration.

Ans: In C language, an array can be declared as follows:



1. Type of data to be stored in array.
2. How many maximum elements needed to be stored
3. Array name (known as identifier)

Q4. Identify the errors in the following code segments. (K.B+U.B+A.B)

Ans:

Sr #	Program	Error
A	int a [] = ({2},{3},{4});	Error expected in array initialization wrong syntax used for initializing array
B	for (int i = 0, i<10, i++) printf ("%d\", i);	Error expected in the syntax of for loop statement terminator is not used. for(int i=0,i<10,i++) **accurate syntax: for(int =0;i<10;i++)**

C	int a [] = {1,2,3,4,5};
	for (int j = 0; j<5; j++)
	printf("%d", a(j));
		Error expected in printf("%d",a(j));
		writing a variable in brackets is not a proper syntax.

D	float f [ ] = {1.4, 3.5, 7.3, 5.9};
	int size =4;
	for (int n = - 1; n< size; n--)
	printf("%f\n", f [n]);
		Error expected in printf("%f\n",f[n]);
		index value of an array cannot be negative.

E	int count = 0;
	for (int i =4; i<6; i--)
	for(int j = i, j<45; j++)
	{
	count++;
	pirntf("%count", count)
	}
		Error expected in printf("%count",count)..
		• %count is not right syntax of integer format specifier.
		• Statement terminator (;) missing.

Q5. Write down output of the following code segments. (K.B+U.B+A.B)

Ans:

Sr #	Program	Output

a	int sum = 0, p;
	for (p = 5; p < = 25; p = p +5)
	sum = sum + 5;
		Sum is 25

b	int i ; for (i = 34; i < = 60; i = i * 2) printf ("* ");	*

c	for (int i = 50; i < = 50; i ++)
	{
	for (j = i; j > = 48; j --)
	printf("j=%d \ n", j);
	printf ("i = % d \ n", i);
	}
		j = 50
		j = 49
		j = 48
		i = 50

d	int i, arr [ ] = {2, 3, 4, 5, 6, 7, 8};	4

	for (int = 0; i < 7; i++)
	{
	printf("%d\n", arr[i] * arr[i]);
	i++;
	}
		16
		36
		64

e	int i, j;
	float ar1[ ] = {1.1, 1.2, 1.3};
	float ar2[ ] = {2.1, 2.2, 2.3};
	for(i = 0; i < 3; i++)
	for (j = i; j<3; j++)
	printf("%f\n", ar1 [i] * ar2 [j] * i * j);
		0.000000
		0.000000
		0.000000
		2.640000
		5.520000
		11.959999

PROGRAMMING EXERCISES (A.B)

EXERCISE 1

Use loops to print following patterns on console.

a)
```
*****
*****
*****
```

b)
```
A
BC
DEF
GHIJ
```

KLMN

(A)

```
# include<stdio.h>
void main ( )
{
int i,j;
for(i=1;i<=3;i++)
{
for(j=1;j<=i;j++)
{
printf(“*****”);
}

printf(“\n”);

}

}
```

(B)

```
#include <stdio.h>
int main( )
{
int i,j,rows=5;
char alphabet='A';
printf("\nHere your pattern\n");
for(i=1; i<=rows; i++)
{
for(j=1; j<=i; j++)

{
printf(alphabet++);
}
printf(“\n”);

}

}
```

EXERCISE 2

Write a program that takes two positive integers a and b as input and displays the value of a^b.

```
# include <stdio.h>
int main ( )
        {
                int a,b,result=1;
                printf("Enter a:");
                scanf ("%d",&a);
                printf("Enter b:");
                scanf ("%d",&b);
                for (int i=1;i<=b;i++)
      {
         result=result*a;
      }
   printf("result=%d^%d=%d",a,b,result);
        }
```

EXERCISE 3

Write a program that takes two numbers as input and displays their Greatest Common Divisor (GCD) using Euclidean method.

```
# include <stdio.h>
void main( )
{
int m, n,r;
printf("Enter-two integer numbers: ");
scanf ("%d %d", &m, &n);
for (;n > 0;)
```

```
{
r = m % n;
m = n;
n = r;
}
printf ("GCD = %d \n",m);
}
```

EXERCISE 4

Write a program to display factorials numbers from 1 to 7. (Hint: Use Nested Loops)

```
# include<stdio.h>
void main ( )
{
int i,j,fact=1;
for(i=1;i<=7;i++)
{
for(j=1;j<=i;j++)
{
fact=fact*j;
}
printf("\n%d",fact);
fact=1;
}
}
```

EXERCISE 5

Write a program that takes 10 numbers as input in an array and displays the product of first and last element on console.

```
# include<stdio.h>
void main ( )
{
int i,arr[10],mult;
printf("Enter 10 elements:");
for(i=0;i<10;i++)
scanf("%d",&arr[i]);
mult=arr[0]*arr[9];
printf("Result of first and last element =%d",mult);
}
```

EXERCISE 6

Write a program that declares and initializes an array of 7 elements and tells how many elements in the array are greater than 10.

```
# include <stdio.h>
void main ( )
{
int counter=0,arr[ ]={3,54,22,67,34,29,19};
for(int i=0;i<7;i++)
{
if(arr[i] > 10)
```

```
counter++;
}
printf("The total number of elements greater than 10 are %d",counter);
}
```

ANSWER KEY

4.1 DATA STRUCTURE

1	2	3
A	A	C

4.1.1 ARRAY

4.1.2 ARRAY DECRORATION

4.1.3 ARRAY INITIALIZATION

4.1.4 ACCESSING ARRAY ELEMENTS

4.1.5 USING VARIABLES AS ARRAY INDEXES.

1	2	3	4
A	A	B	C

4.2 LOOP STRUCTURE

4.2.2 FOR LOOP

4.2.3 NESTED LOOP

4.2.5 ARRAY AND LOOPS

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	B	A	A	A	A	A	A	A	D	A	A	D	C	A
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
C	B	C	C	C	C	B	D	D	C	A	A	C	A	B
31	32	33												
A	A	C												