Q.2 a. Write a program to add two numbers using a temporary variable.

Answer:

```
 #include<stdio.h>
#include<conio.h>
int main ()
{
        int num1, num2;
          clrscr ();
        printf("\n Enter the first number : ");
        scanf ("%d", &num1);
        printf ("\n Enter the second number : ");
        scanf ("%d", &num2);
        num1 = num1 + num2;
        num2 =num1 – num2;
        num1 = num1 – num2;
        printf ("\n The first number is %d", num1);
        printf ("\n The second number is %d", num2);
        return 0;
}
```

b. Explain the basic data types that the C language supports.

Answer:
C language provides very few basic data types. Table 1.2 lists the data type, their size, range, and usage for a C programmer.

Table 1.2 Basic data types in C

Data Type	Size in Bytes	Range	Use
char	1	-128 to 127	To store characters
int	2	-32768 to 32767	To store integer numbers
float	4	3.4E-38 to 3.4E+38	To store floating point numbers
double	8	1.7E-308 to 1.7E+308	To store big floating point numbers

c. Explain Bitwise AND operator and Bitwise OR operator in C.

Answer:
Bitwise AND

The bitwise AND operator (&) is a small version of boolean AND (&&) as it performs operation on bits instead of bytes, chars, integers, etc. When we use the bitwise AND operator, the bit in the first operand is ANDed with the corresponding bit in the second operand. The truth table is the same as we had seen in logical AND operation. That is, the bitwise AND operator compares each bit of its first operand with the corresponding bit of its second operand. If both bits are 1, the corresponding bit in the result is 1 and 0 otherwise. For example,

10101010 & 01010101 = 00000000

In a C program, the & operator is used as follows :

```
int a = 10, b = 20, c = 0;
c=a&b;
```

Bitwise OR

The bitwise OR operator (|) is a small version of the boolean OR (||) as it performs operation on bits instead of bytes, chars, integers, etc. When we use the bitwise OR operator, the bit in the first operand is ORed with the corresponding bit in the second operand. The truth table is the same as we had seen in logical OR operation. That is, the bitwise – OR operator compares each bit of its first operand with the corresponding bit of its second operand. If one or both bits are 1, the corresponding bit in the result is 1 and 0 otherwise. For example,

10101010 & 01010101 =11111111

In a C program, the | operator is used as follows :

```
int a= 10, b = 20, c = 0;
c = a| b
```

Q.3 a. Differentiate between while loop and do-while loop. Also give the syntax of both the loops.

Answer:

The while loop provides a mechanism to repeat one or more statements while a particular condition is true.

Syntax of While Loop

```
Statement x;
While (condition)
{
        Statement block;
}
Statement y;
```

The do-while loop is similar to the while loop. The only difference is that in a do-while loop, the test condition is tested at the end of the loop. Now that the test condition is tested at the end, this clearly means that the body of the loop gets executed at least one time (even if the condition is false).

Syntax of do-while Loop

```
Statement x;
do
{
```

```
        statement block;
} while (condition);

statement y;
```

b. What is the difference between signed and unsigned variables?

Answer:
The difference between signed and unsigned numeric variables is that signed variables can be either negative or positive but unsigned variables can only be positive. By default, C makes a signed variable.

c. Write a program to sum the series 1/2 + 2/3 + ---- + n/n +1)

Answer:

```
 #include<stdio.h>
#include<conio.h>
 main ()
 {
        int n;
        float sum = 0.0, a, i;
        clrscr ();
        printf("\n Enter the value of n: ");
        scanf ("%d", &n);
        for (i = 1.0; i<=n;i++)
                {       a = (float) i/(i +1);
               sum = sum +a;
        }
printf ("\n The sum of series 1/2 + 2/3 + ….+%d/%d = %f", n, n+1, sum); return 0;
 }
```

Q.4 b. Explain briefly different pros and cons of recursion.

Answer:
Advantages
The advantages of using a recursive program include:

- Recursive solutions often tend to be shorter and simpler than non-recursive ones.
- Code is clearer and easier to use.
- Recursion represents the original formula to solve a problem.
- Follows a divide-and-conquer technique to solve problems.
- In some (limited) instances, recursion may be more efficient.

Disadvantages
The drawbacks of using a recursive program include:

- For some programmers and readers, recursion is a difficult concept.
- Recursion is implemented using system stack. If the stack space on the system is limited, recursion to a deeper level will be difficult to implement.

- Aborting a recursive process in midstream is slow and sometimes nasty.
- Using a recursive function takes more memory and time to execute as compared to its non-recursive counterpart.
- It is difficult to find bugs, particularly when using global variables.

c. Consider the linear arrays AAA (5:50), BBB (-5:10) and CCC (18)
(i) Find the number of elements in each array.
(ii) Suppose Base (AAA) = 300 and W=4 words per memory cell for AAA. Find the address of AAA [15] and AAA[35].

Answer:
The number of elements is equal to the length; hence use the formula

Length = UB-LB +1

Accordingly, Length (AAA) = 50-5+1 =46
Length (BBB) = 10-(-5) +1
Length (CCC) = 18-1 +1 = 18
Note that Length (CCC) = UB, since LB = 1.
ii. Use the formula

LOC (AAA[K]) = Base (AAA) + *w*(K-LB)

Hence: (LOC(AAA[15] = 300 +4(15-5) = 340
LOC (AAA[35]) = 300 + 4(35-5) = 420

Q.5 a. Write a program to extract the first N characters of a string.

Answer:

```
#include<stdio.h>
#include<conio.h>
int main ()
{
char str[100], substr[100];
int i = 0, n;
clrscr ();
printf ("\n Enter the string : ");
gets (str);
printf ("\n Enter the number of characters to be copied : ");
scanf ("%d", &n);
i = 0;
while (str [i]  != ' \0 ' && I <n)
}
        substr[i] = str [i];
        I++;
}
substr [i] = ' \0 ';
printf (" \n The substring is : ") ;
```

```
        puts (substr);
        getch ();
        return 0;
}
```

b. What are structures? Illustrate a complex structure with an example.

Answer:

Self-referential structures:- Structures that contain a reference to data of its same type. That is, a self-referential structure, in addition to other data, contains a pointer to a data that is of the same type as that of the structure.

Q.6 a. Write a program to read and display the values of an integer array. Allocate space dynamically for the array.

Answer:

```
#include<stdio.h>
#include<conio.h>
main()
{
        int i, n;
        int *arr;
        clrscr ();
        printf ("\n Enter the number of elements in the array : ");
        scanf ("%d", &n);
        arr = (int *) malloc (n*sizeof(int) );
        if (arr == NULL)
        {
                printf ("\n Memory Allocation Failed");
                exit (0);
        }
        for (i=0; i<n;i++)
        {
                printf ("\n Enter the %dth value of the array : ", i);
                scanf ("%d", &arr[i]);
        }
        printf ("\n The array contains the following values - \n");
        for (i=0; i<n; i++)
                printf ("%d', arr[i]); // another way is to write * (arr +i)
        return 0;
}
```

b. Write the basic steps of a merge sort algorithm. What is the complexity of merge sort?

Answer:

The basic steps of a merge sort algorithm are as follows:

- If the array is of length 0 or 1, then it is already sorted.
- Otherwise, divide the unsorted array into two sub-arrays of about half the size.
- Use merge sort algorithm recursively to sort each sub-array.
- Merge the two sub-arrays to form a single sorted list.

Time complexity is $0(n\log_2(n))$ for both average case and worst case.

Q.7 a. How can the polynomial $6x^3 + 9x^2 + 7x + 1$ be represented in the memory using a linked list?

Answer:

Every individual term in a polynomial consists of two parts, a coefficient and a power. Here, 6, 9, 7 and 1 are the coefficients of the terms that have 3, 2, 1, and 0 as their powers respectively.

Every term of a polynomial can be represented as a node of the linked list. Figure 8.32 shows the linked representation of the terms of the above polynomial.

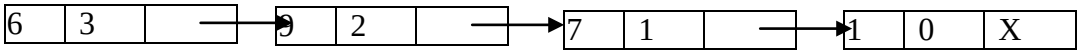


Figure 8.32 Linked representation of a polynomial

b. Write an algorithm to insert an element in a queue.

Answer: Every individual term in a polynomial consists of two parts, a coefficient and a power. Here, 6, 9, 7 and 1 are the coefficients of the terms that have 3, 2, 1, and 0 as their powers respectively.

Every term of a polynomial can be represented as a node of the linked list. Figure 8.32 shows the linked representation of the terms of the above polynomial.

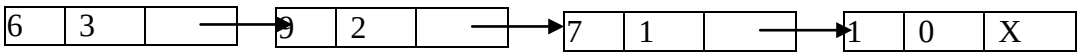


Figure 8.32 Linked representation of a polynomial

c. List few applications of stack.

Answer:

```
Step 1: IF REAR = MAX-1, then;
                Write OVERFLOW
        [END OF IF]
    Step 2: IF FRONT == -1 and REAR = -1, then;
                SET FRONT = REAR = 0
        ELSE
                SET REAR = REAR + 1
```

[END OF IF]
Step 3: SET QUEUE [REAR] = NUM
Step 4: Exit

Q.8 a. Define the following terms with respect to a binary tree:

(i) Degree of a node
(i) In- order traversal
(ii) Depth of the tree
(iv) Full binary tree.

Answer:
i. **Degree** The degree of a node is equal to the number of children that a node has. The degree of a leaf node is zero.

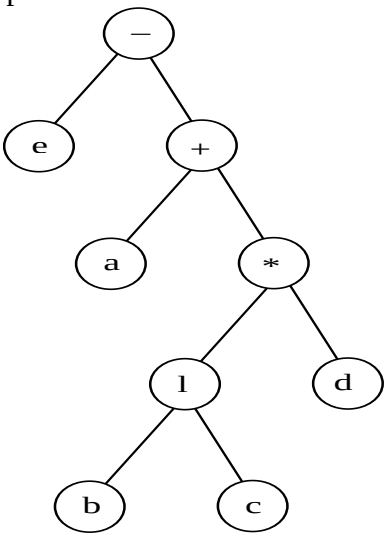
ii. The traversal technique in which all the nodes of a tree are processed by recursively processing the left sub-tree first, then processing the root, and finally the right sub-tree.

iii.The depth of a node N is given as the length of the path from the root R to the node N. The depth of the root node is zero. The height/depth of a tree is defined as the length of the path from the root node to the deepest node in the tree.

iv. **Full binary tree-** A binary tree in which every node has exactly zero or two children.

b. Given the expression, Exp = a+b/c*d-e, construct the corresponding binary tree.

Answer: Expression Tree



c. Write a C routine to find the height of a binary tree.

Answer:

```
int height (node*R)
{
int LH, RH;
if (R = = NULL)
return 0;
else
{
  LH=height (R->left);
 RH = height (R->right);
 if (LH>RH)
 return ++LH;
 else
 return ++RH;
 }
 }
```

Q.9 b. Define the following terms with respect of a graph:

(i) Incident edge
(ii) Degree of vertex
(iii) Directed edge
(iv) Undirected edge
(v) Path

Answer: Page Number 388 of Text Book

c. What is a spanning tree?

Answer:
A minimum weight tree in a weighted graph which contains all of the graph`s vertices.

Text book

C & Data Structures, P.S. Deshpande & O.G. Kakde, Dream tech Press, 2005