Q2 (a) Explain the following operators in C

(i) Increment and decrement operator

(ii) Bitwise operator

(iii) Size of operator

Answer

(i) The increment operator is a unary operator that increases the value of its operand by 1. Similarly, the decrement operator decreases the value of its operand by 1. For example, --x is equivalent to writing x = x - 1.

(ii) Bitwise AND The bitwise AND operator (&) is a small version of the 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 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

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 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

(iii) The operator sizeof is a unary operator used to calculate the size of data types. This operator can be applied to all data types. When using this operator, the keyword sizeof is followed by a type name, variable, or expression. The operator returns the size of the variable, data type, or expression in bytes, i.e., the sizeof operator is used to determine the amount of memory space that the variable/expression/data type will take.

Q2 (b) What are the basic data types that C language supports? Give the size, range and use of each of them.

Answer Basic data types in C

Data type	Keyword used	Size in bytes	Range	Use
Character	char	1	-128 to 127	To store characters
Integer	int	2	-32768 to 32767	To store integer numbers
Floating Point	float	4	3.4E-38 to 3.4E+38	To store floating point numbers
Double	double	8	1.7E-308 to 1.7E+308	To store big floating point numbers
Valueless	void	0	Valueless	---

Q3 (a) Write a program to find whether a given year is a leap year or not.

Answer

```
#include <stdio.h>
#include <conio.h>
int main ( )
{
int year;
clrscr ( );
printf("\n Enter any year: ");
scanf ("%d", &year);
if (year %4 = = 0) && ( ( year%100  !=0) ||
(year%400 = = ) ) )
printf ("\n Leap Year");
else
print f ("\n Not A Leap Year");
return 0;
}
```

Output

```
Enter any year: 1996
Leap Year
```

Q3 (b) Write a program to read the numbers until -1 is encountered. Also count the number of prime numbers and composite numbers entered by the user.

Answer

Write a program using do-while loop to read the numbers until -1 is encountered. Also count the number of prime numbers and composite numbers entered by the user

```
#include <stdio.h>
#include <conio.h>
int main ( )
{
int num, i;
int primes=0, composites=0, flag=0;
clrscr ( );
printf ("\n Enter  -1 to exit… .");
printf ("\n\n enter any number : ) ;
scanf ("%d", &num);
do
{
   for (i=2; i<=num%2;i++)
   {
   if (num% i ==0)
         {
            flag=1;
```

```
            break;
          }
  }
if (flag ==0)
   primes++;
else
   composites++:

flag = 0;
print f ("\n\n Enter any number : ");
scanf ("%d", &num);
} while (num != -1);
print f ("\n count of prime numbers entered = %d", primes);
print f ("\n Count of composite numbers entered = %d", composites);
return 0;
}
```

Q4 (a) What are the advantages and disadvantages of using call-by reference technique of passing arguments?

Answer

Advantages

The advantages of using call-by-reference technique of passing arguments are as follows:

- Since arguments are not copied into new variables, it provides greater time-and space-efficiency.
- The function can change the value of the argument and the change is reflected in the caller.
- A function can return only one value. In case we need to return multiple values, pass those arguments by reference so that modified values are visible in the calling function.

Disadvantages

However, the side-effect of using the technique is that when an argument is passed using call by address, it becomes difficult to tell whether that argument is meant for input, output, or both.

Q4 (b) Write a program / algorithm to merge two integer arrays. Also display the merged array in reverse order.

Answer

```
 # include <stdio.h>
# include <conio.h>
void read_array (int my_array[], int);
void display_array (int my_array[], int);
void merge_array (int my_array3 [], int, int
   my_array1[], int, int my_array2[], int);
void reverse_array (int my_array [], int);
```

```
int main ( )
{
  int arr1[10], arr2 [10], arr3 [20], n, m, t;
  clrscr ( );

  printf ("\n Enter the number of elements in the first array: ");
  scanf ("%d",  &m);
  read_array (arr1, m);
  printf ("\n Enter the number of elements in the second array: ");
  scanf ("%d", &n);
  read_array (arr2, n);
  t = m + n;
  merge_array (arr3, t, arr1, m, arr2, n);

  print f ("\n The merged array in reverse order is :");
  reverse_array (arr3, t);
  getch ( );
  return 0;
}
void read_array (in my_array [10], int n)
{
  int i;
  for (i=0; i<n;i++)
     scanf ("%d", &my_array [i]);
}
  void merge_array (int my_array3 [], int t, int my_array1[], int m, int
my_array2[], int n)
{
  int i, j=0;
  for (i=0; i<m; i++)
  {
     my_array3[j] = my_array1 [i];
      j++;
  }
}
void display_array (int my_array[], int m)
{
   int i;
   for (i = 0; i<n; i++)
      printf ("\n arr[%d] = %d', j,
          my_array [i]);
}
void reverse_array (int my_array[], int m)
{
  int I, j;
  for (i=m-1, j=0; i>=0; i--, j++)
```

```
        printf ("\n arr [%d] = %d", J,
          my_array [i]);
    }
```

Q5 (a) Explain the following string manipulation functions:

(i) strcat function
(ii) strcmp function
(iii) strcpy function

Answer

(i) The strcat function appends the string pointed to by str2 to the end of the string pointed to by str1. The terminating null character of str1 is overwritten. The process stops when the terminating null character of str2 is copied. The argument str1 is returned.

(ii) The strcmp compares the string pointed to by str1 to the string pointed to by str2. The function returns zero if the strings are equal. Otherwise, it returns a value less than zero or greater than zero if str1 is less than or greater than str2 respectively.

(iii) This function copies the string pointed to by str2. It returns the argument str1. Here str1 should be big enough to store the contents of str2.

Q5 (b) Write a program to count the number of lower case numbers, upper case numbers and special characters present in the contents of a file. (Assume that the file contains the following data: 1. Hello, How are you?

Answer

```
#include <stdio.h>
#include <conio.h>
int main (int arg c, char *argv [])
{
   File *fp;
   int ch, upper_case = 0, lower_case = 0.
   numbers = 0, special_chars = 0;
  clrscr ( );
  if (argc ! = 2)
  {
     printf ("\n Full information is not provided");
     return 0;
  }
  fp = fopen (argv [1], "r");
  if (fp = = NULL)
  {

     printf ("\n File Opening Error");
```

```
            return 0;
}
i = 0;
while (feof(fp) = = 0)
{
    fscanf (fp, "%c", &ch);
    if (ch >='A' && ch <= 'Z')
       upper_case++;
    if (ch >= 'a' && ch <='z')
        lowe_case++;
    if (ch >= '0' && ch <= '9')
        numbers++;
    else
        special_chars++;
 }
fclose (fp);
printf ("\n Number of upper case
characters = %d", upper_case);
printf ("\n Number of lower case
characters = %d", lower_case);
printf ("\n Number of digits = %d", numbers);
printf ("\n Number of special characters
         =%d", special_chars);
getch( );
return 0;
}
```

Output

Number of upper case characters = 2
Number of lower case characters = 2
Number of digits = 1
Number of special characters = 2

Q6 (a) Explain Bubble sort. Write an algorithm to sort an array A with N elements.

Answer

In bubble sort, each element is compared with its adjacent element. If the first element is larger than the second one then the position of the elements are interchanged, otherwise it is not changed. Then next element is compared with its adjacent element and the same process is repeated for all the elements in the array. During the pass, the second largest element occupies the second last position. During the next pass, the same process is repeated leaving the largest element. During this pass, the largest element occupies the *n-1* position. The same process is repeated until no more elements are left for comparison. Finally the array is sorted one.

This algorithm sorts the Array ***A*** with ***N*** elements

1. Initialisation
   Set I = 0
2. Repeat steps 3 to 5 until I < N
3. Set J = 0
4. Repeat step 5 until J < N – i – 1
5. If A [J] > A [J + 1] then
   Set temp= A[J]
   Set A [J] = A[J +1]
   Set A[J + 1] = temp
   End If
6. Exit

Q6 (b) Write a program in C that finds transpose of an input matrix.

Answer Page Number 185 of Text-Book

Q7 (a) Write an algorithm to insert a new node at the end of a singly linked list.

Answer

Step 1 : IF AVAIL = NULL, then
Write OVERFLOW
Go to Step 10
[END OF IF]
Step 2 : SET New_Node = AVAIL
Step 3 : SET AVAIL = AVAIL: ->NEXT
Step 4 : SET New_Node-> DATA = VAL
Step 5 :SET New_Node->Next = NULL
Step 6 ;SET PTR = START
Step 7 : Repeat Step 8 while PTR->NEXT ! = NULL
Step 8 : SET PTR = PTR->NEXT
[END OF LOOP]
Step 9 : SET PTR->NEXT = New_Node
Step 10 : EXIT

Q7 (b) Convert the following infix expression into prefix expression.
(A+B) / C(C+D) – (D*E)

Answer

Infix expression : (A + B)/ (C + D) – (D * E)
[+AB] / [+CD] – [*DE]
[/ +AB+CD] – (*DE]
–/+AB+CD*DE

Q7 (c) When an element is added to the deque with n memory cells, what happens to LEFT or RIGHT?

Answer

If the element is added on the left, then LEFT is decreased by 1 (mod n). On the other hand, if the element is added on the right, then RIGHT is increased by 1 (mod n).

Q8 (a) Suppose a binary tree T is in memory. Write a recursive procedure which finds the depth DEP of T.

Answer

The depth DEP of T is 1 more than the maximum of the depths of the left and right subtrees of T. Accordingly:

DEPTH (LEFT, RIGHT, ROOT, DEP)

This procedure finds the depth DEP of a binary tree T in memory.

1. If ROOT = NULL, then: Set DEP := 0, and Return.
2. Call DEPTH (LEFT, RIGHT, LEFT[ROOT], DEPL).
3. Call DEPTH (LEFT, RIGHT, RIGHT[ROOT], DEPR).
4. If DEPL ≥ DEPR, then:
   Set DEP := DEPL + 1.
   Else:
   Set DEP := DEPR + 1.
   [End of If structure.]
5. Return.

Q8 (b) Write an algorithm for post order traversal of a binary tree.

Answer

A binary tree T is in memory. This algorithm does a post order traversal of TR, applying an operation PRO to each of its nodes. An array STK is used to temporarily hold the address of nodes.

1. [Path NULL onto STK and initialize NEXT.]
   Set Top := 1, STK [1] : = NULL, and NEXT := ROOT

2. [Push left-most path onto STK.]
   Repeat Steps 3 to 5 while NEXT … NULL:

3. Set TOP : = TOP + 1 and STACK[TOP] : = NEXT
   [Pushes NEXT on STK.]

4. If RIGHT [NEXT] ≠NULL, then [Push on STK.]
   Set Top : = TOP + 1 and STK [TOP] : = - RCHILD [NEXT].
   [End of If structure.]

1. Set NEXT : = LEFT [NEXT]. [Updates pointer NEXT.]
   [End of Step 2 loop.]

2. Set NEXT : = STACK [TOP] and TOP : = TOP – 1.
   [Pops node from STACK.]

3. Repeat while NEXT > 0:
   Apply PROCESS to INFO[NEXT].
   Set NEXT : = STACK[TOP] and TOP : = TOP – 1.
   [Pops node from STACK.]

4. If NEXT < 0, then:
   Set PTR : = -NEXT.
   Go to Step 2.
   [End of If structure.]

Exit.

Q9 (a) List and explain any four applications of graphs.

Answer
Graphs are constructed for various types of applications such as

- In circuit networks where points of connection are drawn as vertices and component wires become the edges of the graph.
- In transport networks where stations are drawn as vertices and routes become the edges of the graph.
- In maps that draw cities/states/regions as vertices and adjacency relation as edges.
- In program flow analyses, where procedures or modules are treated as vertices and calls to these procedures are drawn as edges of the graph.

Once we have a graph of a particular concept, they can be easily used for finding shortest paths, project planning, etc

Q9 (b) What do you mean by spanning tree and minimum spanning tree? Explain giving a suitable example.

Answer Page Number 413 of Text-Book

Q9 (c) Write an algorithm for DFS traversal. Give an example to justify.

Answer Page Number 396 of Text-Book

Text Book

C & Data Structures, P.S. Deshpande and O.G. Kakde, Dreamtech Press, 2005