ASSIGNMNENT 2nd

[Date]

C Programming language

Adarsh kumar
Section : o
Univ. roll no. : 2115500008

C PROGRAMMING

Cantante:

- If else
- Switch case
- Bitwise operator
- While/ do / for loop
- 1D & 2D array
- String
- function
- Pointers
- DMA(Dynamic Memory Allocation)
- Conditional operators

C PROGRAMMING

If else condition

C PROGRAMMING

```
/*C program to check whether a character is alphabet or not
#include <stdio.h>
int main()
{
   char ch;
   printf("Enter any character: ");
   scanf("%c", &ch);
      if((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z'))
   {
      printf("Character is an ALPHABET.");
   }
   else
   {
      printf("Character is NOT ALPHABET.");
   }
   return 0;
}*/

/*C program to check whether a character is uppercase or lowercase
#include <stdio.h>
int main()
{
   char ch;
   printf("Enter any character: ");
   scanf("%c", &ch);
   if(ch >= 'A' && ch <= 'Z')
   {
      printf("'%c' is uppercase alphabet.", ch);
   }
   else if(ch >= 'a' && ch <= 'z')
   {
      printf("'%c' is lowercase alphabet.", ch);
```

```
    }
    else
    {
        printf("'%c' is not an alphabet.", ch);
    }
    return 0;
} */
```

Q 3. /* C program to count minimum number of notes in an amount

```
#include <stdio.h>
int main()
{
    int amount;
    int note500, note100, note50, note20, note10, note5, note2, note1;
    note500 = note100 = note50 = note20 = note10 = note5 = note2 = note1 = 0;
    printf("Enter amount: ");
    scanf("%d", &amount);
    if(amount >= 500)
    {
        note500 = amount/500;
        amount -= note500 * 500;
    }
    if(amount >= 100)
    {
        note100 = amount/100;
        amount -= note100 * 100;
    }
    if(amount >= 50)
    {
        note50 = amount/50;
        amount -= note50 * 50;
    }
```

```
if(amount >= 20)
{
   note20 = amount/20;
   amount -= note20 * 20;
}
if(amount >= 10)
{
   note10 = amount/10;
   amount -= note10 * 10;
}
if(amount >= 5)
{
   note5 = amount/5;
   amount -= note5 * 5;
}
if(amount >= 2)
{
   note2 = amount /2;
   amount -= note2 * 2;
}
if(amount >= 1)
{
   note1 = amount;
}
printf("Total number of notes = \n");
printf("500 = %d\n", note500);
printf("100 = %d\n", note100);
printf("50 = %d\n", note50);
printf("20 = %d\n", note20);
printf("10 = %d\n", note10);
printf("5 = %d\n", note5);
printf("2 = %d\n", note2);
```

```
    printf("1 = %d\n", note1);


    return 0;
}*/
```

Q 4. / * C program to enter marks of five subjects and find percentage and grade

```
#include <stdio.h>
int main()
{
    int phy, chem, bio, math, comp;
    float per;
    printf("Enter five subjects marks: ");
    scanf("%d%d%d%d%d", &phy, &chem, &bio, &math, &comp);
    per = (phy + chem + bio + math + comp) / 5.0;
    printf("Percentage = %.2f\n", per);
    if(per >= 90)
    {
        printf("Grade A");
    }
    else if(per >= 80)
    {
        printf("Grade B");
    }
    else if(per >= 70)
    {
        printf("Grade C");
    }
    else if(per >= 60)
    {
        printf("Grade D");
    }
    else if(per >= 40)
```

```
    {
        printf("Grade E");
    }
    else
    {
        printf("Grade F");
    }

    return 0;
}*/
```

Q 5. /* C program to calculate gross salary of an employee

```
 #include <stdio.h>
int main()
{
    float basic, gross, da, hra;
    printf("Enter basic salary of an employee: ");
    scanf("%f", &basic);
    if(basic <= 10000)
    {
        da  = basic * 0.8;
        hra = basic * 0.2;
    }
    else if(basic <= 20000)
    {
        da  = basic * 0.9;
        hra = basic * 0.25;
    }
    else
    {
        da  = basic * 0.95;
        hra = basic * 0.3;
    }
```

```
    gross = basic + hra + da;
    printf("GROSS SALARY OF EMPLOYEE = %.2f", gross);
    return 0;
} */
```

C PROGRAMMING

Switch case

C PROGRAMMING

Q.6 /* C program to check vowel or consonant using switch case

```
#include <stdio.h>
int main()
{
   char ch;
   printf("Enter any alphabet: ");
   scanf("%c", &ch);
   switch(ch)
   {
      case 'a':
         printf("Vowel");
         break;
      case 'e':
         printf("Vowel");
         break;
      case 'i':
         printf("Vowel");
         break;
      case 'o':
         printf("Vowel");
         break;
      case 'u':
         printf("Vowel");
         break;
      case 'A':
         printf("Vowel");
         break;
      case 'E':
         printf("Vowel");
         break;
      case 'I':
         printf("Vowel");
```

```
            break;
        case 'O':
            printf("Vowel");
            break;
        case 'U':
            printf("Vowel");
            break;
        default:
            printf("Consonant");
    }
    return 0;
}*/
```

Q.7 /*C program to check positive negative or zero using switch case

```
#include <stdio.h>
int main()
{
    int num;
    printf("Enter any number: ");
    scanf("%d", &num);

    switch (num > 0)
    {
        case 1:
            printf("%d is positive.", num);
        break;
        case 0:
            switch (num < 0)
            {
                case 1:
                    printf("%d is negative.", num);
                    break;
```

```
            case 0:
                printf("%d is zero.", num);
                break;
            }
        break;
    }

    return 0;
} */
```

Q. 8 /* C program to create Simple Calculator using switch case

```
 #include <stdio.h>
int main()
{
    char op;
    float num1, num2, result=0.0f;
    printf("WELCOME TO SIMPLE CALCULATOR\n");
    printf("---------------------------\n");
    printf("Enter [number 1] [+ - * /] [number 2]\n");
    scanf("%f %c %f", &num1, &op, &num2);
    switch(op)
    {
        case '+':
            result = num1 + num2;
            break;
        case '-':
            result = num1 - num2;
            break;
        case '*':
            result = num1 * num2;
            break;
        case '/':
            result = num1 / num2;
```

C PROGRAMMING

```
            break;
        default:
            printf("Invalid operator");
    }
    printf("%.2f %c %.2f = %.2f", num1, op, num2, result);
    return 0;
} */
```

C PROGRAMMING

Bitwise operator

Q. 9 /*C program to find highest order set bit in a number

```
#include <stdio.h>
#define INT_SIZE sizeof(int) * 8
int main()
{
   int num, order = -1, i;
   printf("Enter any number: ");
   scanf("%d", &num);
   for(i=0; i<INT_SIZE; i++)
   {
      if((num>>i) & 1)
         order = i;
   }

   if (order != -1)
      printf("Highest order set bit in %d is %d", num, order);
   else
      printf("0 has no set bits.");

   return 0;
}*/
```

Q.10 /* C program to count total of zeros and ones in a binary number using bitwise operator

```
#include <stdio.h>
#define INT_SIZE sizeof(int) * 8 /* Total number of bits in integer */

int main()
{
   int num, zeros, ones, i;
   printf("Enter any number: ");
   scanf("%d", &num);
```

C PROGRAMMING

```
    zeros = 0;
    ones = 0;
    for(i=0; i<INT_SIZE; i++)
    {
       if(num & 1)
          ones++;
       else
          zeros++;
       num >>= 1;
    }
    printf("Total zero bit is %d\n", zeros);
    printf("Total one bit is %d", ones);
    return 0;
}*/
```

C PROGRAMMING

While/ do / for loop

C PROGRAMMING

Q. 11 /*C program to print all natural numbers from 1 to n

```
#include <stdio.h>
int main()
{
   int i, n;
   printf("Enter any number: ");
   scanf("%d", &n);
   printf("Natural numbers from 1 to %d : \n", n);
 for(i=1; i<=n; i++)
   {
      printf("%d\n", i);
   }
   return 0;
}*/
```

Q. 12 /*C program to find last digit of a number

```
#include <stdio.h>
int main()
{
   int n, lastDigit;
   printf("Enter any number: ");
   scanf("%d", &n);
   lastDigit = n % 10;
   printf("Last digit = %d", lastDigit);
   return 0;
}*/
```

Q. 13 /*C program to find reverse of a number

```
#include <stdio.h>
int main()
{
   int num, reverse = 0;
   printf("Enter any number to find reverse: ");
   scanf("%d", &num);
```

```
    while(num != 0)
    {
        reverse = (reverse * 10) + (num % 10);
        num /= 10;
    }
    printf("Reverse = %d", reverse);
    return 0;
}*/
```

Q. 14 /*C program to count frequency of digits in a given number

```
#include <stdio.h>
#define BASE 10
int main()
{
    long long num, n;
    int i, lastDigit;
    int freq[BASE];
    printf("Enter any number: ");
    scanf("%lld", &num);
    for(i=0; i<BASE; i++)
    {
        freq[i] = 0;
    }
    n = num;
    while(n != 0)
    {
        lastDigit = n % 10;
        n /= 10;
        freq[lastDigit]++;
    }
    printf("Frequency of each digit in %lld is: \n", num);
    for(i=0; i<BASE; i++)
```

```
    {
        printf("Frequency of %d = %d\n", i, freq[i]);
    }


    return 0;
}*/
```

Q. 15 /* C program to calculate factorial of a number

```
#include <stdio.h>
int main()
{
    int i, num;
    unsigned long long fact=1LL;
    printf("Enter any number to calculate factorial: ");
    scanf("%d", &num);
    for(i=1; i<=num; i++)
    {
        fact = fact * i;
    }
    printf("Factorial of %d = %llu", num, fact);
    return 0;
}*/
```

Q.16 /*C program to print Fibonacci series up to n terms

```
 #include <stdio.h>
int main()
{
    int a, b, c, i, terms;
    printf("Enter number of terms: ");
    scanf("%d", &terms);
    a = 0;
    b = 1;
    c = 0;
    printf("Fibonacci terms: \n");
    for(i=1; i<=terms; i++)
```

```
    {
        printf("%d, ", c);
        a = b;
        b = c;
        c = a + b;
    }
    return 0;
}*/
```

Q.17. /*C program to check whether a number is palindrome or not

```
#include <stdio.h>
int main()
{
    int n, num, rev = 0;
    printf("Enter any number to check palindrome: ");
    scanf("%d", &n);
    num = n;
    while(n != 0)
    {
        rev = (rev * 10) + (n % 10);
        n  /= 10;
    }
    if(rev == num)
    {
        printf("%d is palindrome.", num);
    }
    else
    {
        printf("%d is not palindrome.", num);
    }
    return 0;
}*/
```

C PROGRAMMING

1D and 2D array

C PROGRAMMING

Q.18 /*C program to count frequency of each element of array

```
#include <stdio.h>
int main()
{
   int arr[100], freq[100];
   int size, i, j, count;
  printf("Enter size of array: ");
   scanf("%d", &size);
  printf("Enter elements in array: ");
   for(i=0; i<size; i++)
   {
      scanf("%d", &arr[i]);
     freq[i] = -1;
   }
  for(i=0; i<size; i++)
   {
      count = 1;
      for(j=i+1; j<size; j++)
      {
         if(arr[i]==arr[j])
         {
            count++;
            freq[j] = 0;
         }
      }
     if(freq[i] != 0)
      {
         freq[i] = count;
      }
   }
  printf("\nFrequency of all elements of array : \n");
   for(i=0; i<size; i++)
   {
      if(freq[i] != 0)
```

```
        {
            printf("%d occurs %d times\n", arr[i], freq[i]);
        }
    }
    return 0;
}*/
```

Q.19 /*C program to find sum of all elements of array

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
    int arr[MAX_SIZE];
    int i, n, sum=0;
    printf("Enter size of the array: ");
    scanf("%d", &n);
    printf("Enter %d elements in the array: ", n);
    for(i=0; i<n; i++)
    {
        scanf("%d", &arr[i]);
    }
    for(i=0; i<n; i++)
    {
        sum = sum + arr[i];
    }
    printf("Sum of all elements of array = %d", sum);
    return 0;
} */
```

Q. 20 /*C program to find sum of all elements of array

```
 #include <stdio.h>
#define MAX_SIZE 100
int main()
```

```
{
   int arr[MAX_SIZE];
   int i, n, sum=0;
   printf("Enter size of the array: ");
   scanf("%d", &n);
   printf("Enter %d elements in the array: ", n);
   for(i=0; i<n; i++)
   {
      scanf("%d", &arr[i]);
      sum += arr[i];
   }
   printf("Sum of all elements of array = %d", sum);
   return 0;
}*/
```

Q. 21 /*C program to print array in reverse order

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
   int arr[MAX_SIZE];
   int size, i;
  printf("Enter size of the array: ");
   scanf("%d", &size);
  printf("Enter elements in array: ");
   for(i=0; i<size; i++)
   {
      scanf("%d", &arr[i]);
   }
   printf("\nArray in reverse order: ");
   for(i = size-1; i>=0; i--)
   {
      printf("%d\t", arr[i]);
   }
   return 0;
```

```
}
```

Q. 22 /* C program to delete all duplicate elements from array

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
   int arr[MAX_SIZE];
   int size;
   int i, j, k;
   printf("Enter size of the array : ");
   scanf("%d", &size);
   printf("Enter elements in array : ");
   for(i=0; i<size; i++)
   {
      scanf("%d", &arr[i]);
   }
   for(i=0; i<size; i++)
   {
      for(j=i+1; j<size; j++)
      {
         if(arr[i] == arr[j])
         {
            for(k=j; k < size - 1; k++)
            {
               arr[k] = arr[k + 1];
            }
            size--;
            j--;
         }
      }
   }
   printf("\nArray elements after deleting duplicates : ");
   for(i=0; i<size; i++)
   {
```

```
        printf("%d\t", arr[i]);
    }
    return 0;
}*/
```

Q.23 /*C program to insert an element in array at specified position

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
    int arr[MAX_SIZE];
    int i, size, num, pos;
    printf("Enter size of the array : ");
    scanf("%d", &size);
    printf("Enter elements in array : ");
    for(i=0; i<size; i++)
    {
        scanf("%d", &arr[i]);
    }
    printf("Enter element to insert : ");
    scanf("%d", &num);
    printf("Enter the element position : ");
    scanf("%d", &pos);
    if(pos > size+1 || pos <= 0)
    {
        printf("Invalid position! Please enter position between 1 to %d", size);
    }
    else
    {
        for(i=size; i>=pos; i--)
        {
            arr[i] = arr[i-1];
        }
        arr[pos-1] = num;
        size++;
```

```
        printf("Array elements after insertion : ");
        for(i=0; i<size; i++)
        {
            printf("%d\t", arr[i]);
        }
    }
    return 0;
} */
```

Q.24 /*C program to read and print elements in an array

```
#include <stdio.h>
#define MAX_SIZE 1000
int main()
{
    int arr[MAX_SIZE];
    int i, N;
    printf("Enter size of array: ");
    scanf("%d", &N);
    printf("Enter %d elements in the array : ", N);
    for(i=0; i<N; i++)
    {
        scanf("%d", &arr[i]);
    }
    printf("\nElements in array are: ");
    for(i=0; i<N; i++)
    {
        printf("%d, ", arr[i]);
    }
    return 0;
}*/
```

Q.25 /*C program to find sum of all elements of array

```
#include <stdio.h>
#define MAX_SIZE 100

int main()
```

```
{
   int arr[MAX_SIZE];
   int i, n, sum=0;
   printf("Enter size of the array: ");
   scanf("%d", &n);
   printf("Enter %d elements in the array: ", n);
   for(i=0; i<n; i++)
   {
      scanf("%d", &arr[i]);
   }
   for(i=0; i<n; i++)
   {
      sum = sum + arr[i];
   }
   printf("Sum of all elements of array = %d", sum);
   return 0;
}*/
```

Q. 26 /*C program to find maximum and minimum element in array

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
   int arr[MAX_SIZE];
   int i, max, min, size;
   printf("Enter size of the array: ");
   scanf("%d", &size);
   printf("Enter elements in the array: ");
   for(i=0; i<size; i++)
   {
      scanf("%d", &arr[i]);
   }
   max = arr[0];
   min = arr[0];
   for(i=1; i<size; i++)
```

```
    {
        if(arr[i] > max)
        {
            max = arr[i];
        }
        if(arr[i] < min)
        {
            min = arr[i];
        }
    }
    printf("Maximum element = %d\n", max);
    printf("Minimum element = %d", min);
    return 0;
}*/
```

Q. 27 /*C program to find second largest number in an array

```
#include <stdio.h>
#include <limits.h>
#define MAX_SIZE 1000
int main()
{
    int arr[MAX_SIZE], size, i;
    int max1, max2;
    printf("Enter size of the array (1-1000): ");
    scanf("%d", &size);
    printf("Enter elements in the array: ");
    for(i=0; i<size; i++)
    {
        scanf("%d", &arr[i]);
    }
    max1 = max2 = INT_MIN;
    for(i=0; i<size; i++)
    {
        if(arr[i] > max1)
        {
```

```
            max2 = max1;
            max1 = arr[i];
        }
        else if(arr[i] > max2 && arr[i] < max1)
        {
            max2 = arr[i];
        }
    }
    printf("First largest = %d\n", max1);
    printf("Second largest = %d", max2);
    return 0;
}*/
```

Q. 28 /*C program to insert an element in array at specified position

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
    int arr[MAX_SIZE];
    int i, size, num, pos;
    printf("Enter size of the array : ");
    scanf("%d", &size);
    printf("Enter elements in array : ");
    for(i=0; i<size; i++)
    {
        scanf("%d", &arr[i]);
    }
    printf("Enter element to insert : ");
    scanf("%d", &num);
    printf("Enter the element position : ");
    scanf("%d", &pos);
    if(pos > size+1 || pos <= 0)
    {
        printf("Invalid position! Please enter position between 1 to %d", size);
    }
```

```
    else
    {
       for(i=size; i>=pos; i--)
       {
          arr[i] = arr[i-1];
       }
       arr[pos-1] = num;
       size++;
       printf("Array elements after insertion : ");
       for(i=0; i<size; i++)
       {
          printf("%d\t", arr[i]);
       }
    }
    return 0;
}*/
```

C PROGRAMMING

String

C PROGRAMMING

Q.29 /* C program to find length of a string using for loop

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
   char text[MAX_SIZE];
   int i;
   int count= 0;
   printf("Enter any string: ");
   gets(text);
   for(i=0; text[i]!='\0'; i++)
   {
      count++;
   }
   printf("Length of '%s' = %d", text, count);
   return 0;
}*/
```

Q. 30 /*C program to copy one string to another string without using strcpy()

```
#include <stdio.h>
#define MAX_SIZE 100

int main()
{
   char text1[MAX_SIZE];
   char text2[MAX_SIZE];
   int i;
   printf("Enter any string: ");
   gets(text1);
   for(i=0; text1[i]!='\0'; i++)
   {
      text2[i] = text1[i];
   }
  text2[i] = '\0';
```

```
    printf("First string = %s\n", text1);
    printf("Second string = %s\n", text2);
    printf("Total characters copied = %d\n", i);
    return 0;
}*/
```

Q.31 /*C program to compare two string without using string library functions

```
#include <stdio.h>
#define MAX_SIZE 100
int compare(char * str1, char * str2);
int main()
{
    char str1[MAX_SIZE], str2[MAX_SIZE];
    int res;
    printf("Enter first string: ");
    gets(str1);
    printf("Enter second string: ");
    gets(str2);
    res = compare(str1, str2);
    if(res == 0)
    {
        printf("Both strings are equal.");
    }
    else if(res < 0)
    {
        printf("First string is lexicographically smaller than second.");
    }
    else
    {
        printf("First string is lexicographically greater than second.");
    }
    return 0;
}
int compare(char * str1, char * str2)
{
    int i = 0;
    while(str1[i] == str2[i])
    {
        if(str1[i] == '\0' && str2[i] == '\0')
```

```
            break;
        i++;
    }
    return str1[i] - str2[i];
}*/
```

Q. 32 /*C program to check whether a string is palindrome or not

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
    char str[MAX_SIZE];
    int len, startIndex, endIndex;
    printf("Enter any string: ");
    gets(str);
    len = 0;
    while(str[len] != '\0') len++;
    startIndex = 0;
    endIndex   = len-1;
    while(startIndex <= endIndex)
    {
        if(str[startIndex] != str[endIndex])
            break;
        startIndex++;
        endIndex--;
    }
    if(startIndex >= endIndex)
    {
        printf("String is Palindrome.");
    }
    else
    {
        printf("String is Not Palindrome.");
    }
    return 0;
```

```
}*/
```

Q. 33 /*C program to search all occurrences of a character in a string

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
   char str[MAX_SIZE];
   char toSearch;
   int i;
   printf("Enter any string: ");
   gets(str);
   printf("Enter any character to search: ");
   toSearch = getchar();
   i=0;
   while(str[i]!='\0')
   {
      if(str[i] == toSearch)
      {
         printf("'%c' is found at index %d\n", toSearch, i);
      }
      i++;
   }
   return 0;
}*/
```

Q. 34 /*C program to find maximum occurring character in a string

```
#include <stdio.h>
#define MAX_SIZE 100
#define MAX_CHARS 255
int main()
{
   char str[MAX_SIZE];
   int freq[MAX_CHARS];
```

```
    int i = 0, max;
    int ascii;
    printf("Enter any string: ");
    gets(str);
    for(i=0; i<MAX_CHARS; i++)
    {
        freq[i] = 0;
    }
    i=0;
    while(str[i] != '\0')
    {
        ascii = (int)str[i];
        freq[ascii] += 1;
        i++;
    }
    max = 0;
    for(i=0; i<MAX_CHARS; i++)
    {
        if(freq[i] > freq[max])
            max = i;
    }
    printf("Maximum occurring character is '%c' = %d times.", max, freq[max]);
    return 0;
}*/
```

Q. 35 /*C program to remove all repeated characters from a given string

```
#include <stdio.h>
#define MAX_SIZE 100
void removeDuplicates(char * str);
void removeAll(char * str, const char toRemove, int index);
int main()
{
    char str[MAX_SIZE];
    printf("Enter any string: ");
```

```
   gets(str);
   printf("String before removing duplicates: %s\n", str);
   removeDuplicates(str);
   printf("String after removing duplicates: %s\n", str);
   return 0;
}
void removeDuplicates(char * str)
{
   int i = 0;
   while(str[i] != '\0')
   {
      removeAll(str, str[i], i + 1);
      i++;
   }
}
void removeAll(char * str, const char toRemove, int index)
{
   int i;
   while(str[index] != '\0')
   {
         if(str[index] == toRemove)
      {
            i = index;
         while(str[i] != '\0')
         {
            str[i] = str[i + 1];
            i++;
         }
      }
      else
      {
         index++;
      }
   }
}*/
```

C PROGRAMMING

Q. 36 /* C program to remove all repeated characters from a given string

```
#include <stdio.h>
#define MAX_SIZE 100
void removeDuplicates(char * str);
void removeAll(char * str, const char toRemove, int index);
int main()
{
   char str[MAX_SIZE];
   printf("Enter any string: ");
   gets(str);
   printf("String before removing duplicates: %s\n", str);
   removeDuplicates(str);
   printf("String after removing duplicates: %s\n", str);
   return 0;
}
void removeDuplicates(char * str)
{
   int i = 0;
   while(str[i] != '\0')
   {
      removeAll(str, str[i], i + 1);
      i++;
   }
}
void removeAll(char * str, const char toRemove, int index)
{
   int i;
   while(str[index] != '\0')
   {
      if(str[index] == toRemove)
```

```
    {
        i = index;
        while(str[i] != '\0')
        {
            str[i] = str[i + 1];
            i++;
        }
    }
    else
    {
        index++;
    }
  }
}*/
```

Q.37 /*C program to find last occurrence of a word in given string

```
#include <stdio.h>
#include <string.h>
#define MAX_SIZE 100
int main()
{
  char str[MAX_SIZE];
  char word[MAX_SIZE];
  int i, j, found;
  int strLen, wordLen;
  printf("Enter any string: ");
  gets(str);
  printf("Enter any word to search: ");
  gets(word);
  strLen  = strlen(str);
  wordLen = strlen(word);
 for(i=0; i<strLen - wordLen; i++)
```

```
    {
        found = 1;
        for(j=0; j<wordLen; j++)
        {
            if(str[i + j] != word[j])
            {
                found = 0;
                break;
            }
        }
        if(found == 1)
        {
            printf("'%s' found at index: %d \n", word, i);
        }
    }
    return 0;
}*/
```

Q. 38 /*C program to remove extra blank spaces from a given string

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_SIZE 100
char * removeBlanks(const char * str);
int main()
{
    char str[MAX_SIZE];
    char * newString;
    printf("Enter any string: ");
    gets(str);
    printf("\nString before removing blanks: \n'%s'", str);
    newString = removeBlanks(str);
    printf("\n\nString after removing blanks: \n'%s'", newString);
    return 0;
}
```

```
char * removeBlanks(const char * str)
{
   int i, j;
   char * newString;
   newString = (char *)malloc(MAX_SIZE);
   i = 0;
   j = 0;
   while(str[i] != '\0')
   {
      if(str[i] == ' ')
      {
         newString[j] = ' ';
         j++;
         while(str[i] == ' ')
            i++;
      }
      newString[j] = str[i];
      i++;
      j++;
   }
   newString[j] = '\0';
   return newString;
}*/
```

Functions

C PROGRAMMING

Q.39 /*C program to check prime, armstrong and perfect numbers using functions

```
#include <stdio.h>
#include <math.h>
int isPrime(int num);
int isArmstrong(int num);
int isPerfect(int num);
int main()
{
   int num;
   printf("Enter any number: ");
   scanf("%d", &num);
   if(isPrime(num))
   {
      printf("%d is Prime number.\n", num);
   }
   else
   {
      printf("%d is not Prime number.\n", num);
   }
   if(isArmstrong(num))
   {
      printf("%d is Armstrong number.\n", num);
   }
   else
   {
      printf("%d is not Armstrong number.\n", num);
   }
   if(isPerfect(num))
   {
      printf("%d is Perfect number.\n", num);
   }
   else
   {
      printf("%d is not Perfect number.\n", num);
   }
```

```
    return 0;
}
int isPrime(int num)
{
    int i;
    for(i=2; i<=num/2; i++)
    {
        if(num%i == 0)
      {
         return 0;
      }
    }
    return 1;
}
int isArmstrong(int num)
{
    int lastDigit, sum, originalNum, digits;
    sum = 0;
    originalNum = num;
    digits = (int) log10(num) + 1;
    while(num > 0)
    {
       lastDigit = num % 10;
       sum = sum + round(pow(lastDigit, digits));
       num = num / 10;
    }
    return (originalNum == sum);
}
int isPerfect(int num)
{
    int i, sum, n;
    sum = 0;
    n = num;
    for(i=1; i<n; i++)
    {
```

```
    if(n%i == 0)
    {
        sum += i;
    }
  }
  return (num == sum);
}*/
```

Q. 40 /*C program to find maximum and minimum between two numbers using functions

```
#include <stdio.h>
int max(int num1, int num2);
int min(int num1, int num2);
int main()
{
  int num1, num2, maximum, minimum;
  printf("Enter any two numbers: ");
  scanf("%d%d", &num1, &num2);
  maximum = max(num1, num2);
  minimum = min(num1, num2);
  printf("\nMaximum = %d\n", maximum);
  printf("Minimum = %d", minimum);
  return 0;
}
int max(int num1, int num2)
{
  return (num1 > num2 ) ? num1 : num2;
}
int min(int num1, int num2)
{
  return (num1 > num2 ) ? num2 : num1;
} */
```

Q 41. /*C program to print strong numbers in a given interval using functions

```
#include <stdio.h>
long long fact(int num);
void printStrongNumbers(int start, int end);
int main()
```

```
{
   int start, end;
   printf("Enter the lower limit to find strong number: ");
   scanf("%d", &start);
   printf("Enter the upper limit to find strong number: ");
   scanf("%d", &end);
   printf("All strong numbers between %d to %d are: \n", start, end);
   printStrongNumbers(start, end);
   return 0;
}
void printStrongNumbers(int start, int end)
{
   long long sum;
   int num;
   while(start != end)
   {
      sum = 0;
      num = start;
      while(num != 0)
      {
         sum += fact(num % 10);
         num /= 10;
      }
      if(start == sum)
      {
         printf("%d, ", start);
      }
      start++;
   }
}
long long fact(int num)
{
   if(num == 0)
      return 1;
   else
```

```
    return (num * fact(num-1));
}
 #include <stdio.h>
long long fact(int num);
void printStrongNumbers(int start, int end);
int main()
{
   int start, end;
   printf("Enter the lower limit to find strong number: ");
   scanf("%d", &start);
   printf("Enter the upper limit to find strong number: ");
   scanf("%d", &end);

   printf("All strong numbers between %d to %d are: \n", start, end);
   printStrongNumbers(start, end);

   return 0;
}
void printStrongNumbers(int start, int end)
{
   long long sum;
   int num;
   while(start != end)
   {
      sum = 0;
      num = start;
      while(num != 0)
      {
         sum += fact(num % 10);
         num /= 10;
      }
      if(start == sum)
      {
         printf("%d, ", start);
      }
```

```
        start++;
    }
}
long long fact(int num)
{
    if(num == 0)
        return 1;
    else
        return (num * fact(num-1));
}*/
```

Q. 42 /*C program to print all Armstrong numbers between a given range

```
#include <stdio.h>
int isArmstrong(int num);
void printArmstrong(int start, int end);
int main()
{
    int start, end;
    printf("Enter lower limit to print armstrong numbers: ");
    scanf("%d", &start);
    printf("Enter upper limit to print armstrong numbers: ");
    scanf("%d", &end);
    printf("All armstrong numbers between %d to %d are: \n", start, end);
    printArmstrong(start, end);
    return 0;
}
int isArmstrong(int num)
{
    int temp, lastDigit, sum;
    temp = num;
    sum = 0;
    while(temp != 0)
    {
        lastDigit = temp % 10;
        sum += lastDigit * lastDigit * lastDigit;
        temp /= 10;
```

```
    }
    if(num == sum)
        return 1;
    else
        return 0;
}
void printArmstrong(int start, int end)
{
    while(start <= end)
    {
        if(isArmstrong(start))
        {
            printf("%d, ", start);
        }
        start++;
    }
}*/
```

Q. 43 /*C program to print all perfect numbers in given range using function

```
 #include <stdio.h>
int isPerfect(int num);
void printPerfect(int start, int end);
int main()
{
    int start, end;
    printf("Enter lower limit to print perfect numbers: ");
    scanf("%d", &start);
    printf("Enter upper limit to print perfect numbers: ");
    scanf("%d", &end);
    printf("All perfect numbers between %d to %d are: \n", start, end);
    printPerfect(start, end);
    return 0;
}
int isPerfect(int num)
{
    int i, sum;
```

```
    sum = 0;
    for(i=1; i<num; i++)
    {
        if(num % i == 0)
        {
            sum += i;
        }
    }
    if(sum == num)
        return 1;
    else
        return 0;
}
void printPerfect(int start, int end)
{
    while(start <= end)
    {
        if(isPerfect(start))
        {
            printf("%d, ", start);
        }
        start++;
    }
}*/
```

C PROGRAMMING

Pointers

C PROGRAMMING

Q. 44 /* C program to get memory address using address of operator

```
#include <stdio.h>
int main()
{
   char character = 'C';
   int integer = 1;
   float real = 10.4f;
   long long biginteger = 989898989ll;
   printf("Value of character = %c, Address of character = %u\n", character, &character);
   printf("Value of integer = %d, Address of integer = %u\n", integer, &integer);
   printf("Value of real = %f, Address of real = %u\n", real, &real);
   printf("Value of biginteger = %lld, Address of biginteger = %u", biginteger, &biginteger);
   return 0;
}*/
```

Q. 45 /*C program to swap two number using call by reference

```
#include <stdio.h>
void swap(int * num1, int * num2);
int main()
{
   int num1, num2;
   printf("Enter two numbers: ");
   scanf("%d%d", &num1, &num2);
   printf("Before swapping in main n");
   printf("Value of num1 = %d \n", num1);
   printf("Value of num2 = %d \n\n", num2);
   swap(&num1, &num2);
   printf("After swapping in main n");
   printf("Value of num1 = %d \n", num1);
   printf("Value of num2 = %d \n\n", num2);
   return 0;
}
/* Function to swap two numbers*/
void swap(int * num1, int * num2)
```

```
{
   int temp;
  temp = *num1;
  *num1= *num2;
  *num2= temp;
   printf("After swapping in swap function n");
   printf("Value of num1 = %d \n", *num1);
   printf("Value of num2 = %d \n\n", *num2);
}*/
```

Q. 46 /* C program to sort an array using pointers.

```
#include <stdio.h>
#define MAX_SIZE 100
void inputArray(int * arr, int size);
void printArray(int * arr, int size);
int sortAscending(int * num1, int * num2);
int sortDescending(int * num1, int * num2);
void sort(int * arr, int size, int (* compare)(int *, int *));
int main()
{
   int arr[MAX_SIZE];
   int size;
   printf("Enter array size: ");
   scanf("%d", &size);
   printf("Enter elements in array: ");
   inputArray(arr, size);
   printf("\n\nElements before sorting: ");
   printArray(arr, size);
   printf("\n\nArray in ascending order: ");
   sort(arr, size, sortAscending);
   printArray(arr, size);
   printf("\nArray in descending order: ");
   sort(arr, size, sortDescending);
   printArray(arr, size);
   return 0;
}
```

```
void inputArray(int * arr, int size)
{
   int * arrEnd = (arr + size - 1);
   while(arr <= arrEnd)
      scanf("%d", arr++);
}
void printArray(int * arr, int size)
{
   int * arrEnd = (arr + size - 1);
   while(arr <= arrEnd)
      printf("%d, ", *(arr++));
}
int sortAscending(int * num1, int * num2)
{
   return (*num1) - (*num2);
}
int sortDescending(int * num1, int * num2)
{
   return (*num2) - (*num1);
}
void sort(int * arr, int size, int (* compare)(int *, int *))
{
   int * arrEnd  = (arr + size - 1);
   int * curElem = arr;
   int * elemToSort;
   while(curElem <= arrEnd)
   {
      elemToSort = curElem;
      while(elemToSort <= arrEnd)
      {
         if(compare(curElem, elemToSort) > 0)
         {
            *curElem    ^= *elemToSort;
            *elemToSort ^= *curElem;
            *curElem    ^= *elemToSort;
```

```
        }
        elemToSort++;
      }
      curElem++;
   }
}*/
```

Q. 47/*C program to concatenate two strings using pointer

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
   char str1[MAX_SIZE], str2[MAX_SIZE];
   char * s1 = str1;
   char * s2 = str2;
   printf("Enter first string: ");
   gets(str1);
   printf("Enter second string: ");
   gets(str2);
   while(*(++s1));
   while(*(s1++) = *(s2++));
   printf("Concatenated string = %s", str1);
   return 0;
}*/
```

Q.48 /*C program to return multiple value from a function using array.

```
#include <stdio.h>
#define SIZE 10
int * getNEvenNumbers(const int N, int * numbers);
int main()
{
   int evenNumbers[SIZE];
   int i;
   getNEvenNumbers(SIZE, evenNumbers);
   printf("First %d even numbers are: \n", SIZE);
   for (i = 0; i < SIZE; i++)
   {
```

```
        printf("%d ", *(evenNumbers + i));
    }
    return 0;
}
int * getNEvenNumbers(const int N, int * numbers)
{
    int i;
    for (i = 0; i < N; i++)
    {
        *(numbers + i) = 2 * (i + 1);
    }
    return numbers;
}*/
```

C PROGRAMMING

DMA

(Dynamic Memory Allocation)

C PROGRAMMING

Q. 49 .C program to sort number in ascending order by using malloc function. Use free to release memory.

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
   int i,j,temp,n;
   int *p;
   printf("Enter value of n: ");
   scanf("%d",&n);
   p=(int*)malloc(n*sizeof(int));
   printf("Enter valuesn");
   for(i=0;i<n;i++)
      scanf("%d",&p[i]);
   for(i=0;i<n;i++)
   {
      for(j=i+1;j<n;j++)
      {
         if(p[i]>p[j])
         {
            temp=p[i];
            p[i]=p[j];
            p[j]=temp;
         }
      }
   }
   printf("Ascending ordern");
   for(i=0;i<n;i++)
      printf("%dn",p[i]);
   free(p);
   return 0;
}
```

C PROGRAMMING

Conditional operators

C PROGRAMMING

Q. 50 /*C program to find maximum between three numbers using conditional operator

```
#include <stdio.h>
int main()
{
   int num1, num2, num3, max;
   printf("Enter three numbers: ");
   scanf("%d%d%d", &num1, &num2, &num3);
  max = (num1 > num2 && num1 > num3) ? num1 :
        (num2 > num3) ? num2 : num3;
   printf("\nMaximum between %d, %d and %d = %d", num1, num2, num3, max);
   return 0;
} */
```

Q. 51 /*C program to check leap year using conditional operator

```
#include <stdio.h>
int main()
{
   int year;
   printf("Enter any year: ");
   scanf("%d", &year);
   (year%4==0 && year%100!=0) ? printf("LEAP YEAR") :
      (year%400 ==0 ) ? printf("LEAP YEAR") : printf("COMMON YEAR");
   return 0;
}*/
```