**List of C programs**

**I. Simple Programs**   
1 . Find area of a rectangle (M)  
2. Find ASCII value of a character (M)  
3. Convert Celsius to Fahrenheit (M)  
4. Swap value of two variables (M)  
5. Check the given number is odd or even (M)  
6. Check whether a character is vowel or consonant (M)  
7. Find largest among three numbers (M)  
8. Leap year checking (M)  
9. Positive negative checking (M)  
10. Quadratic equation (M)

**II. Programs using Loops**   
11 . Factorial without using function (M)  
12. Table of N and square of N (M)  
13. Calculate x to the power y (M)  
14. Multiplication table (M)  
15. Sum of natural numbers (M)  
16. Fibonacci starting from any two numbers (M)  
17. Upper case to Lower case (M)  
18. Lower to upper (M)  
19. Pascal triangle (M)  
20. LCM & GCD (O)  
21 . Prime numbers between two ranges (M)  
22. Factors of a number (M)  
23. Prime Factors (M)  
24. Bin to dec and oct (M)  
25. Count the number of digit in an integer (M)  
26. Reverse the digits of given number (M)  
27. Number palindrome (M)  
28. Digit summation (M)  
29. Amstrong checking (M)  
30. Make simple calculator in C (M)  
31 . TO FIND SIN(X) USING SINE SERIES (M)  
32. Exponent series (M)  
33. Floyds Triangle (M)

III. **Programs using Arrays**   
34. Fibonacci using array (M)  
35. Largest among N numbers in an array (M)  
36. Smallest among N numbers in an array (M)  
37. Reverse the array elements (M)  
38. Insert an element in an array (M)  
39. Deleting an array element (M)  
40. Transpose of a matrix (O)  
41 . Duplication removal (M)  
42. Linear Search (M)  
43. Binary search (M)  
44. Split the sorted array (M)  
45. Matrix addition (M)

46. Matrix multiplication (M)  
47. Inverse of a 3X3 matrix (M)

IV. **Programs using Functions**   
48. Factorial using function (M)   
49. Min and Max of array (M)  
50. Bubble Sort (M)  
51 . Convert :Bin to dec; dec to bin (M)  
52. Bin to oct; oct to bin (M)  
53. Dec to Hex (M)  
54. Oct to dec; dec to oct (M)  
55. Stack operation using function (M)  
56. Factorial using recursive function (M)  
57. Fibonacci using recursive function (M)  
58. Sum of N numbers using recursion (M)  
59. Reverse the sentence using recursion (M)  
60. Power using recursion (M)  
61 . Towers of Hanoi (O)  
62. Exponent using recursion (M)  
63. GCD using recursion (M)

V. **Programs using Structures**   
64. Student structure (M)  
65. Players structure (O)  
66. Add two polynomials using structures in function

67. Add two distances using structures (M)  
68. Add two complex numbers (M)  
69. Calculate difference between two time period (M)

VI. **Programs using Strings**   
70. Program to Count Blanks,Tabs and Newlines (M)  
71 . Palindrome checking (M)  
72. convert a name into its ascii values. (M)  
73. calculating string length without strlen function (M)  
74. comparing 2 strings without strcmp function (M)  
75. copying one string to another without using strcpy (M)  
76. string concatenation without using strcat function (M)  
77. Pattern replacement (O)  
78. Finding vowels (M)  
79. Sorting in alphabetical order (M)  
80. Searching sub string in a string (M)  
81 . Find the frequency of a character in a string (M)  
82. Remove characters in string except alphabets (M)  
83. Reverse the given string (M)

VII. **Programs using Pointers**  
84. Area of circle using pointers (M)  
85. function pointers (M)  
86. duplication removal using pointers (M)  
87. Sorting integer array using pointers (M)  
88. Sum of array using pointers (M)

89. Count number of words using pointers (M)  
90. Length of a string using pointers (M)  
91 . Reverse the String Using Pointers (M)

**M – Mandatory – have to learn**

**O – Optional – its up to your interest to check**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_PLAYERS 500

#define INPUT\_FILE  "input.txt"

typedef struct {

  char name[20];

  char team[20];

  int  batting\_avg;

} Cricket;

int read\_players(Cricket \*list)

{

  FILE \*in;

  char \*newline;

  int  i;

  if ((in = fopen(INPUT\_FILE, "r")) == NULL)

    return -1;

  for (i = 0;; i++) {

    if (fgets(list[i].name, sizeof list[i].name, in) == NULL)

      break;

    if ((newline = strchr(list[i].name, '\n')) != NULL)

      \*newline = '\0';

    if (fgets(list[i].team, sizeof list[i].team, in) == NULL)

      return -1;

    if ((newline = strchr(list[i].team, '\n')) != NULL)

      \*newline = '\0';

    if (fscanf(in, "%d", &list[i].batting\_avg) != 1)

      return -1;

    fgetc(in);

  }

  fclose(in);

  return i;

}

int cmp\_name(const void \*a, const void \*b)

{

  return strcmp(((Cricket\*)a)->name, ((Cricket\*)b)->name);

}

int cmp\_team(const void \*a, const void \*b)

{

  return strcmp(((Cricket\*)a)->team, ((Cricket\*)b)->team);

}

int cmp\_avg(const void \*a, const void \*b)

{

  if (((Cricket\*)a)->batting\_avg > ((Cricket\*)b)->batting\_avg)

    return -1;

  else if (((Cricket\*)a)->batting\_avg < ((Cricket\*)b)->batting\_avg)

    return 1;

  else

    return 0;

}

int main(void)

{

  Cricket Player[MAX\_PLAYERS];

  int    n;

  int    i;

  if ((n = read\_players(Player)) != -1) {

    puts("SORT BY NAME");

    qsort(Player, n, sizeof Player[0], cmp\_name);

    for (i = 0; i < n; i++) {

      printf("Player: %s\nTeam: %s\nBatting AVG: %d\n\n",

        Player[i].name, Player[i].team, Player[i].batting\_avg);

    }

    puts("SORT BY TEAM");

    qsort(Player, n, sizeof Player[0], cmp\_team);

    for (i = 0; i < n; i++) {

      printf("Player: %s\nTeam: %s\nBatting AVG: %d\n\n",

        Player[i].name, Player[i].team, Player[i].batting\_avg);

    }

    puts("SORT BY BATTING AVG");

    qsort(Player, n, sizeof Player[0], cmp\_avg);

    for (i = 0; i < n; i++) {

      printf("Player: %s\nTeam: %s\nBatting AVG: %d\n\n",

        Player[i].name, Player[i].team, Player[i].batting\_avg);

    }

  }

  return 0;

}

#include<stdio.h>

int main()

{

char line[150];

int i, j;

printf("Enter a string: ");

gets(line);

for(i = 0; line[i] != '\0'; ++i)

{

while (!( (line[i] >= 'a' && line[i] <= 'z') || (line[i] >= 'A' && line[i] <= 'Z') || line[i] == '\0') )

{

for(j = i; line[j] != '\0'; ++j)

{

line[j] = line[j+1];

}

line[j] = '\0';

}

}

printf("Output String: ");

puts(line);

return 0;

}