PROGRAMS FOR PLACEMENT SET 4 - SOLUTIONS

1)

```
#include<stdio.h>
#include<stdlib.h>
int a[20],b[20],c[40];
int m,n,p,val,i,j,key,pos,temp;
/*Function Prototype*/
void create();
void display();
void insert();
void del();
void search();
void merge();
void sort();
int main()
{
int choice;
do{
printf("\n\n--------Menu-----------\n");
printf("1.Create\n");
printf("2.Display\n");
printf("3.Insert\n");
printf("4.Delete\n");
printf("5.Search\n");
printf("6.Sort\n");
printf("7.Merge\n");
printf("8.Exit\n");
printf("-----------------------");
printf("\nEnter your choice:\t");
scanf("%d",&choice);
switch(choice)
{
case 1: create();
break;
case 2:
display();
break;
case 3:
insert();
break;
case 4:
del();
break;
case 5:
search();
```

```
break;
case 6:
sort();
break;
case 7:
merge();
break;
case 8:
exit(0);
break;
default:
printf("\nInvalid choice:\n");
break;
}
}while(choice!=8);
return 0;
}v
oid create() //creating an array
{
printf("\nEnter the size of the array elements:\t");
scanf("%d",&n);
printf("\nEnter the elements for the array:\n");
for(i=0;i<n;i++)
{
scanf("%d",&a[i]);
}
}//end of create()
void display() //displaying an array elements
{
int i;
printf("\nThe array elements are:\n");
for(i=0;i<n;i++){
printf("%d\t",a[i]);
}
}//end of display()
void insert() //inserting an element in to an array
{
printf("\nEnter the position for the new element:\t");
scanf("%d",&pos);
printf("\nEnter the element to be inserted :\t");
scanf("%d",&val);
for(i=n-1;i>=pos;i--)
{
a[i+1]=a[i];
}
a[pos]=val;
```

```
n=n+1;
}//end of insert()
void del() //deleting an array element
{
printf("\nEnter the position of the element to be deleted:\t");
scanf("%d",&pos);
val=a[pos];
for(i=pos;i<n-1;i++)
{
a[i]=a[i+1];
}
n=n-1;
printf("\nThe deleted element is =%d",val);
}//end of delete()
void search() //searching an array element
{
printf("\nEnter the element to be searched:\t");
scanf("%d",&key);
for(i=0;i<n;i++)
{
if(a[i]==key)
{
printf("\nThe element is present at position %d",i);
break;
}
}
if(i==n)
{
printf("\nThe search is unsuccessful");
}
}
}//end of serach()
void sort() //sorting the array elements
{
for(i=0;i<n-1;i++)
{
for(j=0;j<n-1-i;j++) { if(a[j]>a[j+1])
{
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp;
}
}
}
printf("\nAfter sorting the array elements are:\n");
display();
```

```
}//end of sort
void merge() //merging two arrays
{
printf("\nEnter the size of the second array:\t");
scanf("%d",&m);
printf("\nEnter the elements for the second array:\n");
for(i=0;i<m;i++)
{
scanf("%d",&b[i]);
}
for(i=0,j=0;i<n;i++,j++)
{
c[j]=a[i];
}
for(i=0;i<m;i++,j++)
{
c[j]=b[i];
}
p=n+m;
printf("\nArray elements after merging:\n");
for(i=0;i<p;i++)
{
printf("%d\t",c[i]);
}
}/
```

2)

```
#include<stdio.h>

int main()
{
    int arr1[] = {2,3,4,5,1};
    int arr2[] = {2,3,4,5,1};

    int i;

    for(i=0;i<5;i++)
    {
        if(arr1[i]!=arr2[i])
        {
            printf("Arrays not equal");
            return -1;
        }
    }
    printf("arrays equal");
    return 0;
}
```

3)

```c
#include<stdio.h>
int main()
{
    int n;
    printf("enter array size: ");
    scanf("%d",&n);
    int arr[n];
    int i;
    int odd = 0, even = 0;
    printf("Enter array elements: ");
    for(i = 0; i < n; i++)
    {
        scanf("%d",&arr[i]);
    }
    for(i = 0; i < n; i++)
    {
        if(arr[i] % 2 == 1)
            odd++;
        if(arr[i] % 2 == 0)
            even++;
    }
    if(odd == n)
        printf("Odd");
    else
    if(even == n)
        printf("Even");
    else
        printf("Mixed");
    return 0;
}
```

4)

```
import java.util.Arrays;
import java.util.HashSet;

public class Prog3
{
    static void print_missing_elements(int ar[], int start, int end)
    {
    HashSet<Integer> hs = new HashSet<>(); ////create hasset of integers unique values

    for (int i = 0; i < ar.length; i++)
    hs.add(ar[i]);

    for (int i = start; i <= end; i++)
    {
        if (!hs.contains(i))
        {
            System.out.print(i + " "); ////if array does not contain sequence print value
        }
    }
}

public static void main(String[] args)
{
    int arr[] = { 1, 2, 3, 8, 9 };
    int start = 1, end = 5;
    System.out.print("Missing elements : " );
    print_missing_elements(arr, start, end);
    }
}
```

5)

```c
#include <stdio.h>

// returns true if there is triplet with sum equal
// to 'sum' present in A[]. Also, prints the triplet
int find3Numbers(int A[], int arr_size, int sum)
{
    int l, r,i,j,k;

    // Fix the first element as A[i]
    for (i = 0; i < arr_size - 2; i++) {

        // Fix the second element as A[j]
        for (j = i + 1; j < arr_size - 1; j++) {

            // Now look for the third number
            for (k = j + 1; k < arr_size; k++) {
                if (A[i] + A[j] + A[k] == sum) {
                    printf("Triplet is %d, %d, %d",
                        A[i], A[j], A[k]);
                    return 1;
                }
            }
        }
    }

    // If we reach here, then no triplet was found
    return 0;
}

/* Driver program to test above function */
int main()
{
    int A[] = { 1, 4, 45, 6, 10, 8 };
    int sum = 22;
    int arr_size = sizeof(A) / sizeof(A[0]);
    find3Numbers(A, arr_size, sum);
    return 0;
}
```

6)

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

        int array[100], i, num;
        printf("Enter the size of an array \n");

        scanf("%d", &num);
        printf("Enter the elements of the array \n");

        for (i = 0; i < num; i++)
        {
            scanf("%d", &array[i]);
        }

        printf("Even numbers in the array are - ");
        for (i = 0; i < num; i++)
        {
            if (array[i] % 2 == 0)
            {
                printf("%d \t", array[i]);
            }
        }

        printf("\n Odd numbers in the array are -");
        for (i = 0; i < num; i++)
        {
            if (array[i] % 2 != 0)
            {
                printf("%d \t", array[i]);
            }
        }

    }
```

7)

```c
#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;
}
```

8)

```c
#include<stdio.h>

int main()
{
    int a[50],i,n,large,small;
    printf("How many elements:");
    scanf("%d",&n);
    printf("Enter the Array:");

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

    large=small=a[0];
    for(i=1;i<n;++i)
    {
        if(a[i]>large)
            large=a[i];
        if(a[i]<small)
            small=a[i];
    }

    printf("The largest element is %d",large);
    printf("\nThe smallest element is %d",small);

    return 0;
}
```

9)

```c
#include<stdio.h>

int main()
{
    int n,i;
    int sum=0;
    printf("enter number of elements: ");
    scanf("%d",&n);
    int arr[n];
    for(i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    for(i=0;i<n;i++)
    {
        sum +=arr[i];
    }
    printf("sum of array elements is : %d",sum);
    return 0;
}
```

10)

```c
#include<stdio.h>

int check_palindrome(int n)
[
    int div = 1;
    while (n / div >= 10)
    div *= 10;

    while (n != 0)
    {
    int first = n / div;
    int last = n % 10;

    // If first and last digits are not same then return false
    if (first != last)
    return -1;

    // Removing the leading and trailing digits from the number
    n = (n % div) / 10;

    // Reducing divisor by a factor of 2 as 2 digits are dropped
    div = div / 100;
    }
    return 1;
}
int large_palindrome(int A[], int n)
[
int i,j;
// Sort the array
for(i=0; i<=n; i++)
[
    for(j=i;j<= n;j++)
    {
        if(A[i] >A [j])
        {
            int temp = A[i];
            A[i] = A[j];
            A[j] = temp;
        }
    }
}

for(i=0; i<n; i++)
[
    printf("%d ", A[i]);
}
for(i=n-1;i>=0;i--vi)
    {
        if (check_palindrome(A[i]) == 1)
        return A[i];
    }
    return -1;
}
int main()
[
    int a[15], n, i;
    printf("Enter the number of entries: \n");
    scanf("%d", &n);
    printf("Enter the elements: \n");
    for(i=0; i<n; i++)
    scanf("%d", &a[i]);
    printf("\n Largest Palindrome: %d", large_palindrome(a, n));
    return 0;
}
```

11)

```c
#include <stdio.h>

int main()
{
  int n, a[100], b[100], count = 0, c, d;

  printf("Enter number of elements in array\n");
  scanf("%d", &n);

  printf("Enter %d integers\n", n);

  for (c = 0; c < n; c++)
    scanf("%d", &a[c]);

  for (c = 0; c < n; c++)
  {
    for (d = 0; d < count; d++)
    {
      if(a[c] == b[d])
        break;
    }
    if (d == count)
    {
      b[count] = a[c];
      count++;
    }
  }

  printf("Array obtained after removing duplicate elements:\n");

  for (c = 0; c < count; c++)
    printf("%d\n", b[c]);

  return 0;
```

12)

```c
// C Program to find the minimum scalar product of two vectors (dot product)
#include<stdio.h>

int sort(int arr[], int n)
{
int i, j;
for (i = 0; i < n-1; i++)
for (j = 0; j < n-i-1; j++)
if (arr[j] > arr[j+1])
{
int temp = arr[j];
arr[j] = arr[j+1];
arr[j+1] = temp;
}
}

int sort_des(int arr[], int n)
{
int i,j;
for (i = 0; i < n; ++i)
{
for (j = i + 1; j < n; ++j)
{
if (arr[i] < arr[j])
{
int a = arr[i];
arr[i] = arr[j];
arr[j] = a;
}
}
}
}

int main()
{
//fill the code;
int n;
scanf("%d",&n);
int arr1[n], arr2[n];
int i;
for(i = 0; i < n ; i++)
{
scanf("%d",&arr1[i]);
}
for(i = 0; i < n ; i++)
{
scanf("%d",&arr2[i]);
}

sort(arr1, n);
sort_des(arr2, n);
int sum = 0;
for(i = 0; i < n ; i++)
{
sum = sum + (arr1[i] * arr2[i]);
}
printf("%d",sum);
return 0;
```

13)

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

    int total;
    int i;
    int positiveSum = 0;
    int negativeSum = 0;
    printf("How many numbers you want to add : ");
    scanf("%d",&total);
    int numbers[total];
    for(i=0; i<total; i++){
        printf("Enter number %d : ",(i+1));
        scanf("%d",&numbers[i]);
    }
    for(i=0 ; i<total ; i++){
       if(numbers[i] < 0){
         negativeSum += numbers[i];
       }else{
         positiveSum += numbers[i];
       }
    }
    printf("You have entered : \n");
    for(i=0 ; i<total; i++){
     printf("%d ",numbers[i]);
    }
    printf("\nPositive numbers sum : %d",positiveSum);
    printf("\nNegative numbers sum : %d\n",negativeSum);

}
```

14)

```c
#include<stdio.h>

void main()
{
    int a[50];
    int n,i,small,s_small;

    printf("\n Enter number of elements: ");
    scanf("%d",&n);

    printf("\n Enter %d elements: ",n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }

    small=s_small=a[0];

    for(i=1;i<n;i++)
    {
        if(small>a[i])
        {
            s_small=small;
            small=a[i];
        }
        else if(s_small>a[i] && a[i]!=small)
        {
            s_small=a[i];
        }
    }

    printf("\n The Second Smallest Element in the given Array: %d", s_small);
}
```

15)

```c
#include <stdio.h>
void main()
    {
        int i, j, a, n, number[30];
        printf("Enter the value of N \n");
        scanf("%d", &n);

        printf("Enter the numbers \n");
        for (i = 0; i < n; ++i)
            scanf("%d", &number[i]);
        for (i = 0; i < n; ++i)
        {
            for (j = i + 1; j < n; ++j)
            {
                if (number[i] > number[j])
                {
                    a =  number[i];
                    number[i] = number[j];
                    number[j] = a;

                }
            }
        }
        printf("The numbers arranged in ascending order are given below \n");
        for (i = 0; i < n; ++i)
            printf("%d\n", number[i]);

    }
~
```

16)

```c
#include <stdio.h>

int main()
{
   int n, c, d, a[100], b[100];

   printf("Enter the number of elements in array\n");
   scanf("%d", &n);

   printf("Enter array elements\n");

   for (c = 0; c < n ; c++)
      scanf("%d", &a[c]);

   /*
    * Copying elements into array b starting from end of array a
    */

   for (c = n - 1, d = 0; c >= 0; c--, d++)
      b[d] = a[c];


   for (c = 0; c < n; c++)
      a[c] = b[c];

   printf("Reverse array is\n");

   for (c = 0; c < n; c++)
      printf("%d\n", a[c]);

   return 0;
}
```

17)

```c
// C program to find Maximum Product Subarray
#include <stdio.h>

int min (int x, int y) {return x < y? x : y; }

int max (int x, int y) {return x > y? x : y; }

int maxSubarrayProduct(int arr[], int n)
{
    int i;
    int max_ending_here = 1;

    int min_ending_here = 1;

    int max_so_far = 1;

    for (i = 0; i < n; i++)
    {
        if (arr[i] > 0)
        {
            max_ending_here = max_ending_here*arr[i];
            min_ending_here = min (min_ending_here * arr[i], 1);
        }

        else if (arr[i] == 0)
        {
            max_ending_here = 1;
            min_ending_here = 1;
        }

        else
        {
            int temp = max_ending_here;
            max_ending_here = max (min_ending_here * arr[i], 1);
            min_ending_here = temp * arr[i];
        }

        if (max_so_far < max_ending_here)
        max_so_far = max_ending_here;
    }

    return max_so_far;
}

int main()
{
    int arr[] = {1, -2, -3, 0, 7, -8, -2};
    int n = sizeof(arr)/sizeof(arr[0]);
    printf("Maximum Sub array product is %d",
            maxSubarrayProduct(arr, n));
    return 0;
}
```

18)

```c
#include<stdio.h>

// Returns true if set1[] and set2[] are disjoint, else false
int areDisjoint(int set1[], int set2[], int m, int n)
{
    int i,j;
    // Take every element of set1[] and search it in set2
    for (i=0; i<m; i++)
      for (j=0; j<n; j++)
         if (set1[i] == set2[j])
            return 0;

    // If no element of set1 is present in set2
    return 1;
}

// Driver program to test above function
int main()
{
    int set1[] = {12, 34, 11, 9, 3};
    int set2[] = {7, 2, 1, 5};
    int m = sizeof(set1)/sizeof(set1[0]);
    int n = sizeof(set2)/sizeof(set2[0]);
    areDisjoint(set1, set2, m, n)? printf("Yes"): printf(" No");
    return 0;
}
```

19)

```c
#include<stdio.h>

/* Return 1 if arr2[] is a subset of
arr1[] */
int isSubset(int arr1[], int arr2[],  int m, int n)
{
    int i = 0;
    int j = 0;
    for (i = 0; i < n; i++)
    {
        for (j = 0; j < m; j++)
        {
            if(arr2[i] == arr1[j])
                break;
        }

        /* If the above inner loop was
        not broken at all then arr2[i]
        is not present in arr1[] */
        if (j == m)
            return 0;
    }

    /* If we reach here then all
    elements of arr2[] are present
    in arr1[] */
    return 1;
}

// Driver code
int main()
{
    int arr1[] = {11, 1, 13, 21, 3, 7};
    int arr2[] = {11, 3, 7, 1};

    int m = sizeof(arr1)/sizeof(arr1[0]);
    int n = sizeof(arr2)/sizeof(arr2[0]);

    if(isSubset(arr1, arr2, m, n))
        printf("arr2[] is subset of arr1[] ");
    else
        printf("arr2[] is not a subset of arr1[]");
    return 0;
}
```

20)

```c
#include<stdio.h>

int max(int a,int b)
{
    if (a>b)
        return a;
    else return b;

}

long MaxDotProduct(int A[], int B[], int m, int n)
{
    int i,j;
    long long int dp[n+1][m+1];

    for (i=1; i<=n; i++)

        for (j=i; j<=m; j++)

            dp[i][j] = max((dp[i-1][j-1] + (A[j-1]*B[i-1])) ,
                            dp[i][j-1]);

    return dp[n][m] ;
}
int main()
{
    int A[] = { 2, 3 , 1, 7, 8 } ;
    int B[] = { 3, 6, 7 } ;
    int m = sizeof(A)/sizeof(A[0]);
    int n = sizeof(B)/sizeof(B[0]);
    printf("%ld",MaxDotProduct(A, B, m, n));
    return 0;
}
```

21)

```
#include <stdio.h>

int make_equal(int a[], int n)
{
int i;
int flag = 1;
for (i = 0; i < n; i++)
{
// Divide number by 2
while (a[i] % 2 == 0)
a[i] /= 2;
// Divide number by 3
while (a[i] % 3 == 0)
a[i] /= 3;
}

// Remaining numbers
for (i = 1; i < n; i++)
{
if (a[i] != a[0])
{
flag = 0 ;
}
return flag;
}
}

int main()
{
int n, i;
scanf("%d", &n);
int a[n];
for(i=0; i<n; i++)
scanf("%d", &a[i]);
if (make_equal(a, n) == 1)
printf("Yes");
else
printf("No");
return 0;
}
```

22)

```java
import java.util.HashMap;

class Prog22 {

    // Print all pairs that have a symmetric counterpart
    static void findSymPairs(int arr[][])
    {
        // Creates an empty hashMap hM
        HashMap<Integer, Integer> hM = new HashMap<Integer, Integer>();

        // Traverse through the given array
        for (int i = 0; i < arr.length; i++)
        {
            // First and second elements of current pair
            int first = arr[i][0];
            int sec   = arr[i][1];

            // Look for second element of this pair in hash
            Integer val = hM.get(sec);

            // If found and value in hash matches with first
            // element of this pair, we found symmetry
            if (val != null && val == first)
               System.out.println("(" + sec + ", " + first + ")");

            else  // Else put sec element of this pair in hash
               hM.put(first, sec);
        }
    }

    // Drive method
    public static void main(String arg[])
    {
        int arr[][] = new int[5][2];
        arr[0][0] = 11; arr[0][1] = 20;
        arr[1][0] = 30; arr[1][1] = 40;
        arr[2][0] = 5;  arr[2][1] = 10;
        arr[3][0] = 40;  arr[3][1] = 30;
        arr[4][0] = 10;  arr[4][1] = 5;
        findSymPairs(arr);
    }
}
```

23)

```c
#include <stdio.h>

int countDistinct(int arr[], int n)
{
    int i;
    int res = 1;

    // Pick all elements one by one
    for  (i = 1; i < n; i++) {
        int j = 0;
        for (j = 0; j < i; j++)
            if (arr[i] == arr[j])
                break;

        // If not printed earlier, then print it
        if (i == j)
            res++;
    }
    return res;
}

// Driver program to test above function
int main()
{
    int arr[] = { 12, 10, 9, 45, 2, 10, 10, 45 };
    int n = sizeof(arr) / sizeof(arr[0]);
    printf("%d",countDistinct(arr, n));
    return 0;
}
~
```

24)

```c
#include <stdio.h>

int countDistinct(int arr[], int n)
{
    int i;
    int res = 1;

    // Pick all elements one by one
    for  (i = 1; i < n; i++) {
        int j = 0;
        for (j = 0; j < i; j++)
            if (arr[i] == arr[j])
                break;

        // If not printed earlier, then print it
        if (i == j)
            res++;
    }
    return res;
}

// Driver program to test above function
int main()
{
    int arr[] = { 12, 10, 9, 45, 2, 10, 10, 45 };
    int n = sizeof(arr) / sizeof(arr[0]);
    printf("%d",countDistinct(arr, n));
    return 0;
}
~
```

25)

```c
#include<stdio.h>
void printRepeating(int arr[], int size)
{
    int i, j;
    printf(" Repeating elements are ");
    for(i = 0; i < size; i++)
        for(j = i + 1; j < size; j++)
        if(arr[i] == arr[j])
            printf("%d ",arr[i]);
}

// Driver Code
int main()
{
    int arr[] = {4, 2, 4, 5, 2, 3, 1};
    int arr_size = sizeof(arr)/sizeof(arr[0]);
    printRepeating(arr, arr_size);
}
```