

```

1: //Set Operations on Arrays
2:
3: struct Array
4: {
5:     int A[10];
6:     int size;
7:     int length;
8: };
9:
10: void Display(struct Array arr)
11: {
12:     int i;
13:     printf("\nElements are\n");
14:     for(i=0;i<arr.length;i++)
15:         printf("%d ",arr.A[i]);
16: }
17:
18: struct Array* Union(struct Array *arr1,struct Array *arr2)
19: {
20:     int i,j,k;
21:     i=j=k=0;
22:
23:     struct Array *arr3=(struct Array *)malloc(sizeof(struct Array));
24:
25:     while(i<arr1->length && j<arr2->length)
26:     {
27:         if(arr1->A[i]<arr2->A[j])
28:             arr3->A[k++]=arr1->A[i++];
29:         else if(arr2->A[j]<arr1->A[i])
30:             arr3->A[k++]=arr2->A[j++];
31:         else
32:         {
33:             arr3->A[k++]=arr1->A[i++];
34:             j++;
35:         }
36:     }
37:
38:     for(;i<arr1->length;i++)
39:         arr3->A[k++]=arr1->A[i];

```

```

40:  for(;j<arr2->length;j++)
41:  arr3->A[k++]=arr2->A[j];
42:
43:  arr3->length=k;
44:  arr3->size=10;
45:
46:  return arr3;
47: }
48:
49: struct Array* Intersection(struct Array *arr1,struct Array* arr2)
50: {
51:  int i,j,k;
52:  i=j=k=0;
53:
54:  struct Array *arr3=(struct Array *)malloc(sizeof(struct Array));
55:
56:  while(i<arr1->length && j<arr2->length)
57:  {
58:  if(arr1->A[i]<arr2->A[j])
59:  i++;
60:  else if(arr2->A[j]<arr1->A[i])
61:  j++;
62:  else if(arr1->A[i]==arr2->A[j])
63:  {
64:  arr3->A[k++]=arr1->A[i++];
65:  j++;
66:  }
67:  }
68:
69:  arr3->length=k;
70:  arr3->size=10;
71:
72:  return arr3;
73: }
74:
75: struct Array* Difference(struct Array *arr1,struct Array* arr2)
76: {
77:  int i,j,k;
78:  i=j=k=0;

```

```

79:
80: struct Array *arr3=(struct Array *)malloc(sizeof(struct Array));
81:
82: while(i<arr1->length && j<arr2->length)
83: {
84: if(arr1->A[i]<arr2->A[j])
85: arr3->A[k++]=arr1->A[i++];
86: else if(arr2->A[j]<arr1->A[i])
87: j++;
88: else
89: {
90: i++;
91: j++;
92: }
93: }
94: for(;i<arr1->length;i++)
95: arr3->A[k++]=arr1->A[i];
96:
97:
98: arr3->length=k;
99: arr3->size=10;
100:
101: return arr3;
102: }
103:
104: int main()
105: {
106: struct Array arr1={{2,9,21,28,35},10,5};
107: struct Array arr2={{2,3,9,18,28},10,5};
108: struct Array *arr3;
109: arr3=Union(&arr1,&arr2);
110: Display(*arr3);
111: arr3=Intersection(&arr1,&arr2);
112: Display(*arr3);
113: arr3=Difference(&arr1,&arr2);
114: Display(*arr3);
115:
116: return 0;
117: }

```