# AOA QBANK SOLUTION

```
//BUBBLE SORT
#include <iostream>

using namespace std;

int main()
{   int n;
    cout<<"Enter size of numbers: ";
    cin>>n;
    int arr[n],temp;
    cout<<"Enter array:\n";
    for(int i=0;i<n;i++)
    {
        cin>>arr[i];
    }
    for(int i=0;i<n;i++)
    {
        for(int j=i;j<n;j++)
        {
        if(arr[i]>arr[j])
        {
            temp=arr[i];
            arr[i]=arr[j];
            arr[j]=temp;
        }

        }
    }
    cout<<"Sorted array is:";
    for(int i=0;i<n;i++)
    {
        cout<<arr[i];
        cout<<" ";
    }

    return 0;
```

```
}
```

**//selection sort**
**As simple as me - 🙂..smallest from unsorted , swap it with i**

```
#include <iostream>
//selection sort
using namespace std;
int main ()
{
  int min, n;
  cout << "Enter size of numbers: ";
  cin >> n;
  int arr[n], temp;
  cout << "Enter array:\n";
  for (int i = 0; i < n; i++)
    {
      cin >> arr[i];
    }
  for (int i = 0; i <= n-2; i++)
    {
        min=i;
        for(int j=i;j<=n-1;j++)
        {
            if(arr[j]<arr[min])
            {
                min=j;
            }
            temp=arr[min];
            arr[min]=arr[i];
            arr[i]=temp;
        }

    }
  cout << "Sorted array is:";
  for (int i = 0; i < n; i++)
    {
      cout << arr[i];
      cout << " ";
    }
```

```
  return 0;
}
```

**//Insertion Sort**
**A loop to iterate and another loop to compare 1st nd aage ke elements**

```
/******************************************************************************

Welcome to GDB Online.
GDB online is an online compiler and debugger tool for C, C++, Python, Java, PHP, Ruby, Perl,
C#, OCaml, VB, Swift, Pascal, Fortran, Haskell, Objective-C, Assembly, HTML, CSS,
JS, SQLite, Prolog.
Code, Compile, Run and Debug online from anywhere in world.

*******************************************************************************/
#include <iostream>
//selection sort
using namespace std;
int main ()
{
  int j, n;
  cout << "Enter size of numbers: ";
  cin >> n;
  int arr[n], temp;
  cout << "Enter array:\n";
  for (int i = 0; i < n; i++)
    {
      cin >> arr[i];
    }
  for (int i = 0; i <= n-1; i++)
    {
        j=i;
        while(j>0 && arr[j-1]>arr[j])
        {

            temp=arr[j-1];
            arr[j-1]=arr[j];
            arr[j]=temp;
```

```
            j--;
        }

    }
  cout << "Sorted array is:";
  for (int i = 0; i < n; i++)
    {
      cout << arr[i];
      cout << " ";
    }

  return 0;
}
```

**//MERGE —>**

**Basics vector: vector <int> vect(n);  for loop cin>>a[i]**

```
#include <iostream>
#include<vector>
using namespace std;
void merge (vector < int >&a, int low, int mid, int high)
{
  vector < int >temp;
  int left = low;
  int right = mid + 1;
  while (left <= mid && right <= high)
    {
      if (a[left] <= a[right])
          {
            temp.push_back(a[left]);
            left++;
          }
      else
          {
            temp.push_back(a[right]);
            right++;
          }
    }
    while(left<=mid)
    {
       temp.push_back(a[left]);
```

```
        left++;
    }
    while(right<=high)
    {
        temp.push_back(a[right]);
        right++;
    }
    for(int i=low ;i<=high;i++)
    {
        a[i]=temp[i-low];
    }
}

void mergeS (vector < int >&a, int low,int high)
{
  if (low >= high)
    return;
  int mid = (low + high) / 2;

  {
    mergeS (a, low, mid);
    mergeS (a, mid + 1, high);
    merge (a, low, mid, high);
  }
}

int main ()
{

  int low, high, n,value;
  cout<<"Enter size of array";
  cin>> n;
  vector <int> a(n);
  low = a[0];
  high = a[n - 1];
  for (int i = 0; i < n; i++)
    { cout<<"enter value\n";
      cin>>a[i];
    }
  mergeS (a, 0, n-1);
```

```
  cout<<"Sorted array:";
   for (int i = 0; i < n; i++)
    {
      cout<<a[i];
      cout<<" ";
    }

  return 0;

}
```

**Prims**

```
#include <stdio.h>
int visited[10]={0};
int cost[10][10];
int main()
{
    int n,i,j,ne=1,a=0,b=0,u=0,v=0,min,mincost=0;
    printf("Enter number of nodes");
    scanf("%d",&n);
    printf("Adjacency matrix:\n");
    for(i=1;i<=n;i++)
    {
        for(j=1;j<=n;j++)
        {
            scanf("%d",&cost[i][j]);
            if(cost[i][j]==0)
            cost[i][j]=999;
        }
    }
    visited[1]=1;
    while(ne<n)
    {
        for(i=1,min=999;i<=n;i++)
        {
            if(visited[i]==1)
            {
            for(j=1;j<=n;j++)
            {
                if(visited[j]==0 && cost[i][j]<min)
                {
                    min=cost[i][j];
                    a=u=i;
                    b=v=j;
```

```
            }
          }
          }
       }
       if(visited[u]==0 || visited[v]==0)
       {
          printf("\n %d \t Edge:%d %d\t cost:%d",ne++,a,b,min);
          mincost=mincost+min;
          visited[v]=1;
       }
       cost[u][v]=cost[v][u]=999;
   }
printf("total: %d",mincost);
   return 0;
}
```

```
Enter number of nodes5
Adjacency matrix:
0 2 0 6 0
2 0 3 8 5
0 3 0 0 7
6 8 0 0 9
0 5 7 9 0

 1      Edge:1 2       cost:2
 2      Edge:2 3       cost:3
 3      Edge:2 5       cost:5
 4      Edge:1 4       cost:6total: 16
```

**Knapsack:**

```
#include<stdio.h>
int m, n;
float x[10];
struct knap
{
  int w, p, obj_no;
  float pw;
};
struct knap a[10];
void sortk()
```

```
{
    struct knap temp;
    for(int i=0;i<n;i++)
    {
        for(int j=i+1;j<n;j++)
        {
            if(a[i].pw < a[j].pw)
            {
                temp=a[i];
                a[i]=a[j];
                a[j]=temp;
            }
        }
    }
}
int greedknap()
{
    int i,rem_cap,profit=0;
    rem_cap=m;
    for(i=0;i<n;i++)
    {
        x[i]=0;
    }

    for(i=0;i<n;i++)
    {
        if(a[i].w>rem_cap)
            break;
        profit=profit+a[i].p;
        x[a[i].obj_no-1]=1;
        rem_cap=rem_cap-a[i].w;
    }
    if(i<=n-1)
    {
        x[a[i].obj_no-1]=(float)(rem_cap)/(float)(a[i].w);
        profit=profit+ x[a[i].obj_no-1]*a[i].p;
    }
    return profit;
}
int main ()
{
  int i, value;
  printf ("Enter mass of knap:\n");
  scanf ("%d", &m);
```

```
  printf ("Enter number of items:\n");
  scanf ("%d", &n);
  for (i = 0; i < n; i++)
    {
      printf ("Enter weight of knap:%d\n", i + 1);
      scanf ("%d", &a[i].w);
      a[i].obj_no=i+1;
    }
  for (i = 0; i < n; i++)
    {
      printf ("Enter profit of knap:%d\n", i + 1);
      scanf ("%d", &a[i].p);
    }
  for (i = 0; i < n; i++)
    {
      a[i].pw = (float) (a[i].p) / (float) (a[i].w);
    }
  printf ("Before sorting:\n");
  printf ("item\t\tweight\t\tprofit\t\tratio\n");
  for (i = 0; i < n; i++)
    {

      printf ("%d\t\t%d\t\t%d\t\t%.2f\n", i + 1, a[i].w, a[i].p, a[i].pw);

    }
  printf ("After sorting:\n");
  sortk ();
  printf ("item\t\tweight\t\tprofit\t\tratio\n");
  for (i = 0; i < n; i++)
    {

      printf ("%d\t\t%d\t\t%d\t\t%.2f\n", i + 1, a[i].w, a[i].p, a[i].pw);

    }
    value=greedknap();
    for(i=0;i<n;i++)
    {
        printf("%.2f\t",x[i]);
    }
    printf("\nProfit:%d",value);
}
```

**//LCS**

```
#include<stdio.h>
#include<string.h>
char x[30],y[30];
char b[30][30];
int c[30][30],count=0;
void print_lcs(int i,int j)
{
    if(i==0 || j==0)
    return;
    else if(b[i][j]=='d')
   {
    print_lcs(i-1,j-1);
    count++;
    printf("%c\t",x[i-1]);
   }
   else if(b[i][j]=='t')
   {
      print_lcs(i-1,j);
   }
   else
   {
      print_lcs(i,j-1);
   }
}
void Length(char x[30],char y[30])
{

    int m=strlen(x);
    int n=strlen(y);
    b[m+1][n+1];
    c[m+1][n+1];
    for(int i=0;i<=m;i++)
    {
       c[i][0]=0;
    }
    for(int j=0;j<=n;j++)
    {
       c[0][j]=0;
    }
    for(int i=1;i<=m;i++)
    {
       for(int j=1;j<=n;j++)
       {
          if(x[i-1]==y[j-1])
```

```
            {
                c[i][j]=c[i-1][j-1]+1;
                b[i][j]='d';
            }
            else if(c[i-1][j]>=c[i][j-1])
            {
                c[i][j]=c[i-1][j];
                b[i][j]='t';
            }
            else
            {
                c[i][j]=c[i][j-1];
                b[i][j]='r';
            }
        }
    }
    printf("Matrix b:\n");
    for(int i=0;i<=m;i++)
    {
        for(int j=0;j<=n;j++)
        {
            printf("%c\t",b[i][j]);
        }
        printf("\n");
    }
    printf("Matrix c:\n");
    for(int i=0;i<=m;i++)
    {
        for(int j=0;j<=n;j++)
        {
            printf("%c\t",c[i][j]);
        }
        printf("\n");
    }
    printf("LCS IS:\n");
    print_lcs(m,n);
}
int main()
{
    printf("Enter string 1:\n");
    scanf("%s",x);
    printf("Enter String 2:\n");
    scanf("%s",y);
    printf("String 1 as follows:\n");
```

```
    for(int i=0;i<strlen(x);i++)
    {
        printf("%c\n",x[i]);
    }
    printf("String 2 as follows:\n");
    for(int i=0;i<strlen(y);i++)
    {
        printf("%c\n",y[i]);
    }
    Length(x,y);
    printf("\nTotal length:\t:%d",count);
    return 0;
}
```

**Floyyddwarshall**

```
// Online C compiler to run C program online
#include <stdio.h>
int w[10][10],p[10][10],d[10][10];
void read(int v)
{
    for(int i=0;i<v;i++)
    {
        for(int j=0;j<v;j++)
        {
            scanf("%d",&w[i][j]);
        }
    }
}
void print(int v,int q[10][10])
{
    for(int i=0;i<v;i++)
    {
        for(int j=0;j<v;j++)
        {
            printf("%d\t",q[i][j]);
        }
        printf("\n");
    }
}
```

```
void initialise(int v)
{
   for(int i=0;i<v;i++)
   {
      for(int j=0;j<v;j++)
      {
         if(i==j)
         {
           p[i][j]=-1;
           d[i][j]=0;
         }
         else if(w[i][j]==0)
         {
            p[i][j]=-1;
            d[i][j]=999;
         }
         else
         {
            p[i][j]=i;
            d[i][j]=w[i][j];
         }
      }
   }
}
void fw(int v)
{
   for(int k=0;k<v;k++)
   {
      for(int i=0;i<v;i++)
      {
         for(int j=0;j<v;j++)
         {
            if(d[i][j]<=d[i][k]+d[k][j])
            {
               continue;
            }
            else
            {
               p[i][j]=p[k][j];
               d[i][j]=d[i][k]+d[k][j];
```

```
            }
          }
        }
    }
}
void printpath(int s,int d,int v)
{
    if(s==d)
    {
        printf("%d",s);
    }
    else if(p[s][d]==-1)
    {
        printf("No paths\n");
    }
    else
    {
        printpath(s,p[s][d],v);
        printf("---->%d",d);
    }
}
int main() {
    // Write C code here

    int v;
    printf("Enter number of vertices:\n");
    scanf("%d",&v);
    printf("Enter Adjacent matrix:\n");
    read(v);
    printf("Entered matrix is:\n");
    print(v,w);
    initialise(v);
    printf("After initialising:\nP:\n");
    print(v,p);
    printf("After initialising:\nD:\n");
    print(v,d);
    fw(v);
    printf("After floydd warshall:\n");
    printf("D matrix\n");
    print(v,d);
```

```
    printf("P matrix\n");
    print(v,p);

    printf("\n\nPrinting paths:");
    for(int i=0;i<v;i++)
    {
        for(int j=0;j<v;j++)
        {
            if(i!=j){
            printf("The path from %d to %d is:",i,j);
            printpath(i,j,v);
            printf("\n");
            }
            }
    }

    return 0;
}
```

👑 **Queen**

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
int x[30], count = 0;
int
place (int k, int i)
{
  for (int j = 1; j <= k - 1; j++)
    {
      if (x[j] == i || abs (x[j] - i) == abs (j - k))
          return 0;
    }
  return 1;

}

void
prin1D (int n)
{
  for (int i = 1; i <= n; i++)
```

```
    {
      printf ("%d\t", x[i]);
    }
  printf ("\n");
}

void
printSol (int n)
{
  count++;
  printf ("Solution:%d\n", count);
  for (int i = 1; i <= n; i++)
    {
      for (int j = 1; j <= n; j++)
          {
            if (x[i] == j)
              printf ("Q ");
            else
              printf ("* ");

          }
      printf ("\n");
    }
  printf ("\n");
}

void
Queen (int k, int n)
{
  for (int i = 1; i <= n; i++)
    {
      if (place (k, i))
          {
            x[k] = i;
            if (k == n)
              {
                printSol (n);
                prin1D (n);
              }
            else
```

```
            {
              Queen (k + 1, n);
            }
         }
    }
}

void
main ()
{
  int n;
  printf ("Enter number of queens:\n");
  scanf ("%d", &n);
  Queen (1, n);
}
```

## SUM OF SUBSET

```
#include<stdio.h>

#include<stdlib.h>

int count=0;

void printSub(int subset[],int subsetSize)

{

   for(int i=0;i<subsetSize;i++)

   {

      printf("%d ",subset[i]);

   }

   printf("\n");

}

void subsetSum(int arr[],int subset[],int n,int subsetSize,int sum,int targetSum)

{

   count++;

   if(sum==targetSum)
```

```
    {
        printSub(subset,subsetSize);
        return;
    }
    if(n==0 || sum>targetSum)
    {
        return ;
    }
    subset[subsetSize]=arr[n-1];
    subsetSum(arr,subset,n-1,subsetSize+1,sum+arr[n-1],targetSum);
    subsetSum(arr,subset,n-1,subsetSize,sum,targetSum);
}

int main()
{
    int n;
    printf("Enter number of elements:\n");
    scanf("%d",&n);
    int arr[n];
    printf("Enter elements:\n");
    for(int i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    int targetSum;
    printf("Enter target sum:\n");
    scanf("%d",&targetSum);
    int subset[n];
    subsetSum(arr,subset,n,0,0,targetSum);
```

```
    printf("Number of calls:%d\n",count);
    return 0;
}
```

**//RABIN KARP**

```
#include <stdio.h>
#include <string.h>
#define d 256
void rabin_karp(char* text, char* pattern, int q)
{
    int n=strlen(text);
    int m=strlen(pattern);
    int i,j;
    int p=0;
    int t=0;
    int h=1;
    int found=0;
    for(i=0;i<m-1;i++)
        h=(h*d)%q;
    for(i=0;i<m;i++)
    {
        p=(d*p+pattern[i])%q;
        t=(d*t+text[i])%q;
    }
    for(i=0;i<=n-m;i++)
    {
        if(p==t)
        {
            for(j=0;j<m;j++)
            {
                if(text[i+j]!=pattern[j])
                    break;
            }
            if(j==m)
```

```
            {
                printf("Pattern found at index %d\n", i);
                found=1;
            }
        }
        if(i<n-m)
        {
            t=(d*(t-text[i]*h)+text[i+m])%q;
            if(t<0)
                t=(t+q);
        }
    }
    if(!found)
        printf("Pattern not found in the text!\n");
}
int main()
{
    char text[100],pattern[100];
    int q=101;
    printf("Enter the text: ");
    scanf("%s",text);
    printf("Enter the pattern: ");
    scanf("%s",pattern);
    rabin_karp(text,pattern,q);
    return 0;
}
```