NAME – AYUSH KUSHARE

ROLL NO – 224103305

BRANCH – MECHANICAL (FTE)

COMPUTATIONAL FLUID DYNAMICS

ASSIGNMENT 1

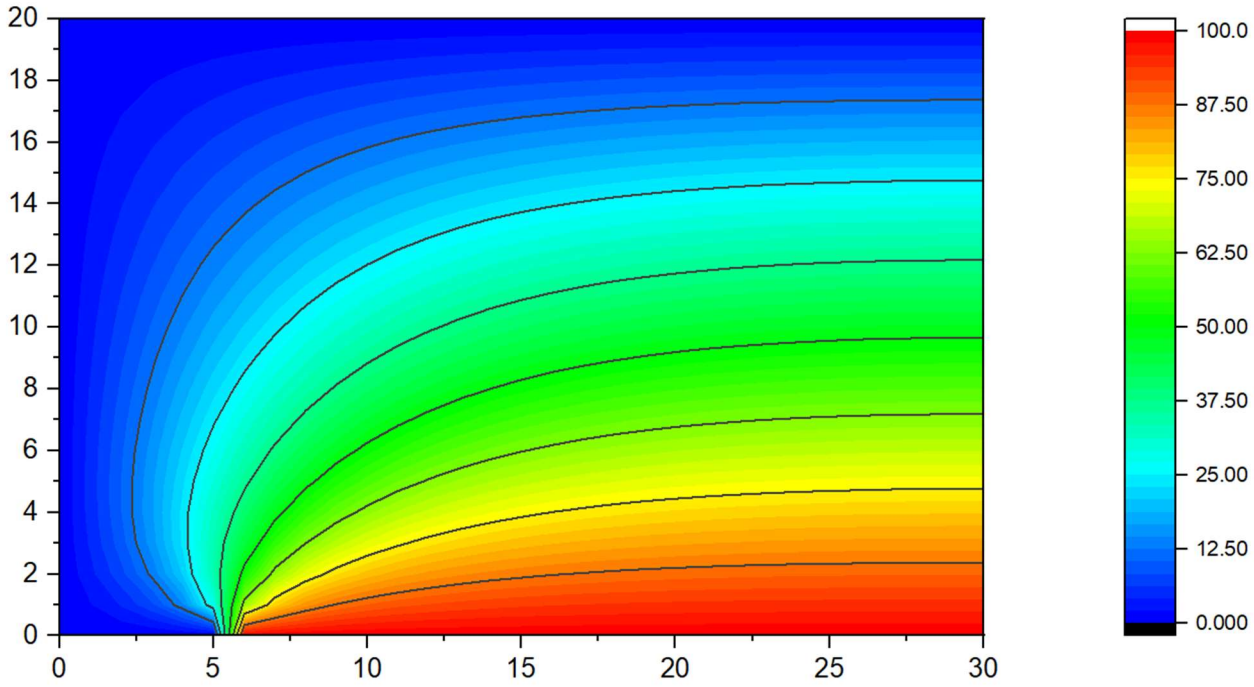
1) JACOBI ITERATIVE METHOD

Problem_1

```
#include<iostream>
#include<math.h>
#include<stdio.h>
using namespace std;
int main()
{
  int m=31,n=21, i ,j;
  double B,dx,dy,s;
  double mat1[100][100],mat2[100][100];
  dx=0.2;
  dy=0.2;
  B=dx/dy;
  s=pow(B,2);
  double l=1/(2*(1+s));
for(i=0;i<m;i++)
{
for (j = 0; j<n; j++)
  {
  if (j == (n-1))
   {
mat2[i][j] = 0.0;
   }
  else if (i == 0)
   {
  mat2[i][j] = 0.0;
   }
  else if (i <= 5)
  {
  mat2[i][j] = 0.0;
  }
  else if (i >= 6)
  {
  mat2[i][j] = 100.0;
  }
  else
  {
  mat2[i][j] = 0.0;
  }
 }
}
 int iteration = 0;
```

```
 double error = 1.0;
do
{
for (i = 0; i < m; i++)
 {
 for (j = 0; j<n; j++)
  {
    mat1[i][j] = mat2[i][j];
  }
 }
  for (i = 1; i < (m-1); i++)
  {
    for (j = 1; j < (n-1); j++)
    {

     mat2[i][j]= l*(s*mat1[i][j-1] + mat1[i-1][j]+mat1[i+1][j] + s*mat1[i][j+1]);
    }
  }
for(j=0;j<n;j++)
 {
  mat2[m-1][j]=mat2[m-2][j];
 }
 error=0;
 for(i=0;i<m;i++)
 {
    for(j=0;j<n;j++)
    {
        error= error + pow(mat2[i][j]-mat1[i][j],2);
    }
 }
 error=sqrt(error/(m*n));

iteration++;

} while(error>0.000001);
for(i=0;i<m;i++)
{
  for(j=0;j<n;j++)
  cout<<i*dx<<"\t"<<j*dy<<"\t"<<mat2[i][j]<<endl;
}
cout<<"no. of iterations = "<<iteration;
return 0;
}
```

OUTPUT : No. of Iterations = 1827



Problem_2

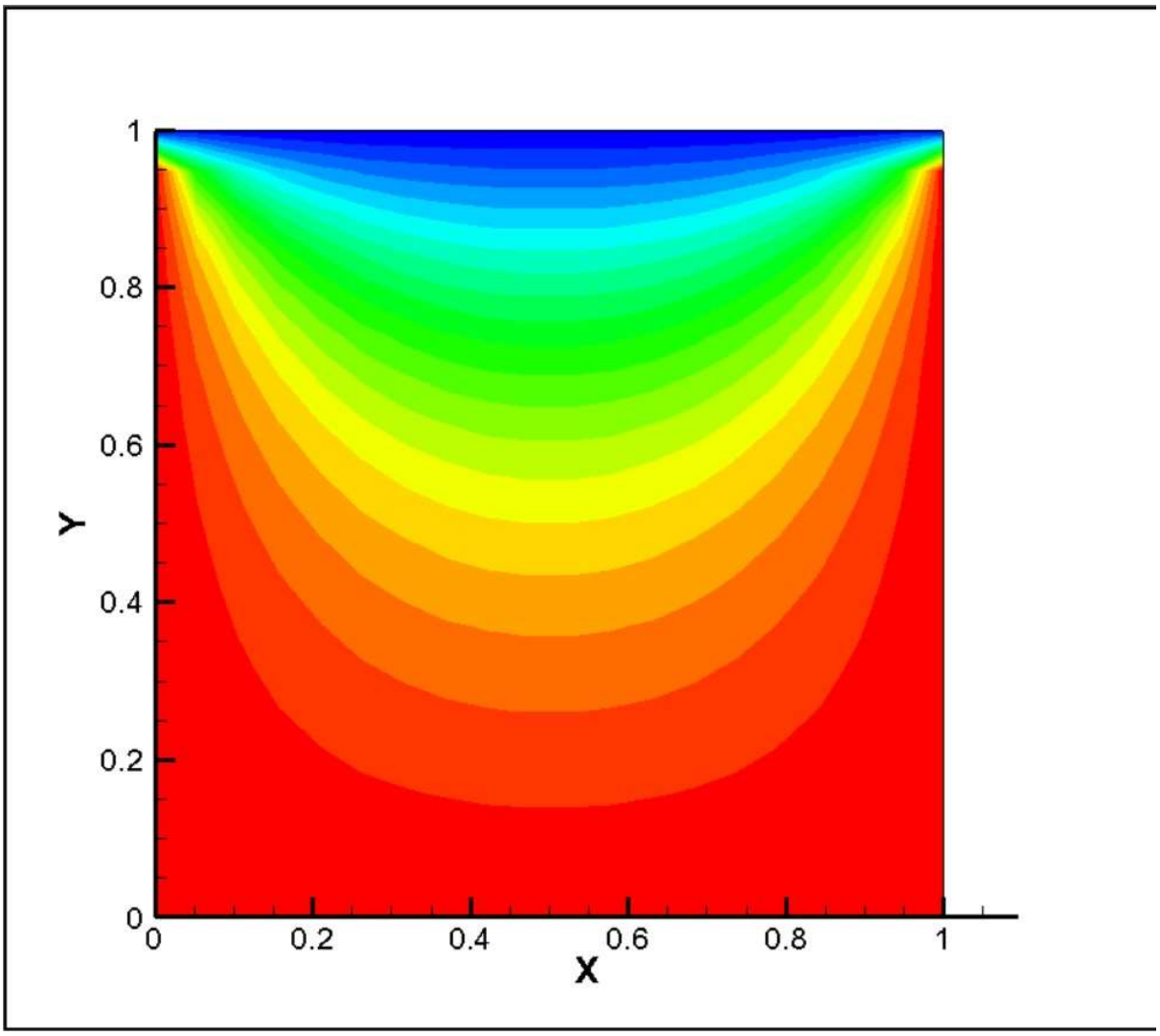
The following program represents the jacobi iteration method. The following program takes grid size m*n as input and calculates the grid size based on the input. The program gives us the values at specific grid points which is plotted using tecplot software.

```
#include<iostream>
#include<math.h>
#include<stdio.h>
using namespace std;
int main()
{
  int m,n, i ,j;
  double B,dx,dy,s;
  double mat1[100][100],mat2[100][100];
   cout<<"enter the size m*n\n";
  cin>>m>>n;
```

```
  dx=1.0/(m-1);
  dy=1.0/(n-1);
  B=dx/dy;
  s=pow(B,2.0);
  double l=1.0/(2.0*(1.0+s));
for(i=0;i<m;i++)
{
for (j = 0; j<n; j++)
  {
      if (j == (n-1))
       {
        mat2[i][j] = 0.0;
       }
      else if (i == 0)
       {
      mat2[i][j] = 1.0;
       }
      else if (j==0)
      {
      mat2[i][j] = 1.0;
      }
      else if(i== (m-1))
      {
        mat2[i][j]=1.0;
      }
      else
      {
      mat2[i][j] = 0.0;
      }
 }
}
 int iteration = 0;
 double error = 1.0;
do
{
for (i = 0; i < m; i++)
 {
 for (j = 0; j<n; j++)
  {
    mat1[i][j] = mat2[i][j];
  }
 }
  for (i = 1; i < (m-1); i++)
  {
    for (j = 1; j < (n-1); j++)
```

```
    {

     mat2[i][j]= l*(s*mat1[i][j-1] + mat1[i-1][j]+mat1[i+1][j] + s*mat1[i][j+1]);
    }
  }
 error=0.0;
 for(i=0;i<m;i++)
 {
    for(j=0;j<n;j++)
    {
        error= error + pow(mat2[i][j]-mat1[i][j],2);
    }
 }
 error=sqrt(error/(m*n));

iteration++;
} while(error>0.000001);
for(i=0;i<m;i++)
{
  for(j=0;j<n;j++)
  cout<<i*dx<<"\t"<<j*dy<<"\t"<<mat2[i][j]<<endl;
}
cout<<"no. of iterations = "<<iteration;
return 0;
}
```

OUTPUT : No. of Iterations = 654


1
0.8
0.6
0.4
0.2
0
Y
0
0.2
0.4
0.6
0.8
1
X

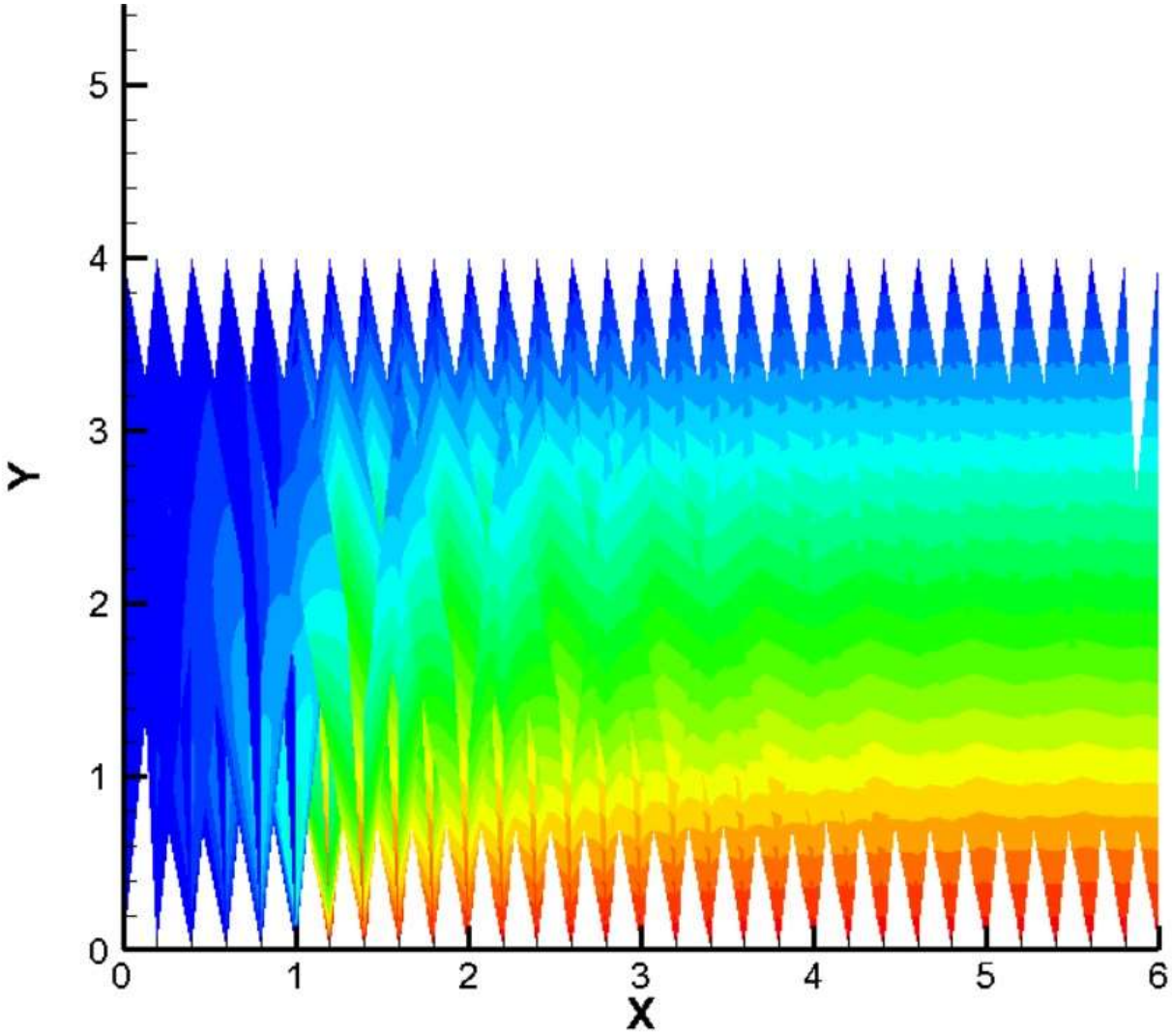
2) POINT GAUSS-SEIDEL ITERATIVE METHOD

Problem_1

```
#include<iostream>
#include<math.h>
#include<stdio.h>
using namespace std;
int main()
{
  int m=31,n=21, i ,j;
  double B,dx,dy,s;
  double mat1[100][100],mat2[100][100];
  dx=0.2;
  dy=0.2;
  B=dx/dy;
  s=pow(B,2);
  double l=1/(2*(1+s));
for(i=0;i<m;i++)
{
for (j = 0; j<n; j++)
  {
  if (j == (n-1))
   {
    mat2[i][j] = 0.0;
   }
  else if (i == 0)
   {
  mat2[i][j] = 0.0;
   }
  else if (i <= 5)
  {
  mat2[i][j] = 0.0;
  }
  else if (i >= 6)
  {
  mat2[i][j] = 100.0;
  }
  else
  {
  mat2[i][j] = 0.0;
  }
 }
```

```
}
 int iteration = 0;
 double error = 1.0;
do
{
for (i = 0; i < m; i++)
 {
 for (j = 0; j<n; j++)
  {
    mat1[i][j] = mat2[i][j];
  }
 }
  for (i = 1; i < (m-1); i++)
  {
    for (j = 1; j < (n-1); j++)
    {
     mat2[i][j]= l*(s*mat2[i][j-1]+mat2[i-1][j]+mat1[i+1][j]+s*mat1[i][j+1]);
    }
  }
for(j=0;j<n;j++)
 {
  mat2[m-1][j]=mat2[m-2][j];
 }
 error=0;
 for(i=0;i<m;i++)
 {
    for(j=0;j<n;j++)
    {
        error= error + pow(mat2[i][j]-mat1[i][j],2);
    }
 }
 error=sqrt(error/(m*n));

iteration++;

} while(error>0.000001);
for(i=0;i<m;i++)
{
  for(j=0;j<n;j++)
  cout<<i*dx<<"\t"<<j*dy<<"\t"<<mat2[i][j]<<endl;
}
cout<<"no. of iterations = "<<iteration;
return 0;
}
```

OUTPUT : No. of Iterations = 982

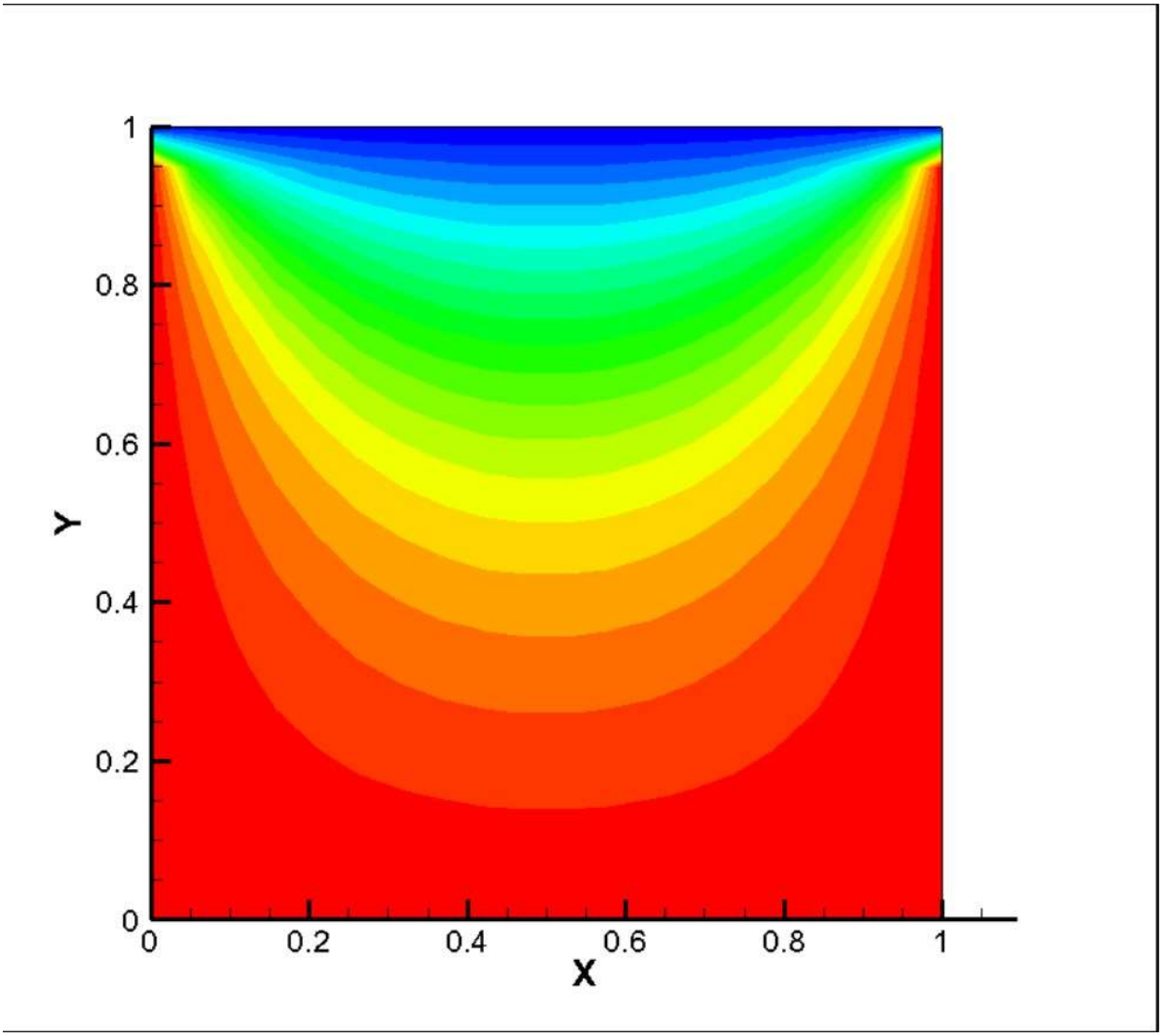


Problem_2

```
#include<iostream>
#include<math.h>
#include<stdio.h>
using namespace std;
int main()
{
  int m,n, i ,j;
  double B,dx,dy,s;
  double mat1[100][100],mat2[100][100];
   cout<<"enter the size m*n\n";
  cin>>m>>n;
  dx=1.0/(m-1);
  dy=1.0/(n-1);
  B=dx/dy;
  s=pow(B,2.0);
  double l=1.0/(2.0*(1.0+s));
for(i=0;i<m;i++)
{
for (j = 0; j<n; j++)
  {
      if (j == (n-1))
       {
        mat2[i][j] = 0.0;
       }
      else if (i == 0)
       {
      mat2[i][j] = 1.0;
       }
      else if (j==0)
      {
      mat2[i][j] = 1.0;
      }
      else if(i== (m-1))
      {
        mat2[i][j]=1.0;
      }
      else
      {
      mat2[i][j] = 0.0;
      }
 }
}
 int iteration = 0;
```

```
 double error = 1.0;
do
{
for (i = 0; i < m; i++)
 {
 for (j = 0; j<n; j++)
  {
    mat1[i][j] = mat2[i][j];
  }
 }
  for (i = 1; i < (m-1); i++)
  {
    for (j = 1; j < (n-1); j++)
    {
      mat2[i][j]= l*(s*mat2[i][j-1]+mat2[i-1][j]+mat1[i+1][j]+s*mat1[i][j+1]);
    }
  }
 error=0.0;
 for(i=0;i<m;i++)
 {
    for(j=0;j<n;j++)
    {
        error= error + pow(mat2[i][j]-mat1[i][j],2);
    }
 }
 error=sqrt(error/(m*n));

iteration++;
} while(error>0.000001);
for(i=0;i<m;i++)
{
  for(j=0;j<n;j++)
  cout<<i*dx<<"\t"<<j*dy<<"\t"<<mat2[i][j]<<endl;
}
cout<<"no. of iterations = "<<iteration;
return 0;
}
```

OUTPUT : No. of Iterations = 352


1
0.8
0.6
0.4
0.2
0
Y
0
0.2
0.4
0.6
0.8
1
X

3) POINT SUCESSIVE OVER RELAXATION (PSOR) METHOD

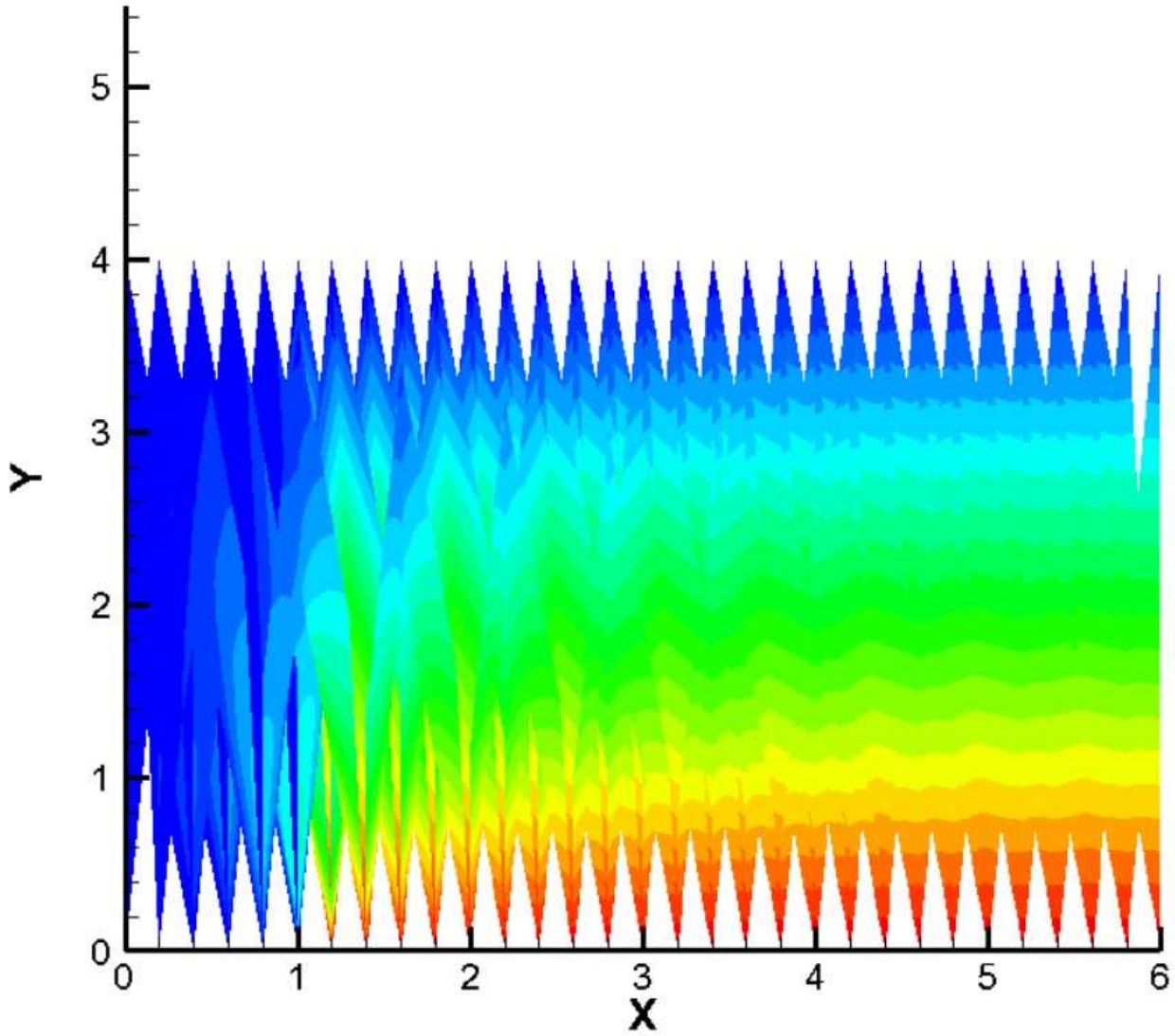
Problem_1

```
#include<iostream>
#include<math.h>
#include<stdio.h>
using namespace std;
int main()
{
  int m=31,n=21, i ,j;
  double B,dx,dy,s, w,a,expression,pi=3.1417;
  double mat1[100][100],mat2[100][100];
  dx=0.2;
  dy=0.2;
  B=dx/dy;
  s=pow(B,2);
  double l=1/(2*(1+s));
  expression=((cos(pi/(m-1))+s*(cos(pi/(n-1))))/(1+s));
   a=pow(expression,2);
   w=(2-sqrt(1-a))/a;
for(i=0;i<m;i++)
{
for (j = 0; j<n; j++)
  {
  if (j == (n-1))
   {
    mat2[i][j] = 0.0;
   }
  else if (i == 0)
   {
  mat2[i][j] = 0.0;
   }
  else if (i <= 5)
  {
  mat2[i][j] = 0.0;
  }
  else if (i >= 6)
  {
  mat2[i][j] = 100.0;
  }
```

```
  else
  {
  mat2[i][j] = 0.0;
  }
 }
}
 int iteration = 0;
 double error = 1.0;
do
{
for (i = 0; i < m; i++)
 {
 for (j = 0; j<n; j++)
  {
    mat1[i][j] = mat2[i][j];
  }
 }
  for (i = 1; i < (m-1); i++)
  {
    for (j = 1; j < (n-1); j++)
    {

      mat2[i][j]= (1-w)*mat1[i][j]+w*l*(s*mat2[i][j-1] + mat2[i-
1][j]+mat1[i+1][j] + s*mat1[i][j+1]);
    }
  }
for(j=0;j<n;j++)
 {
  mat2[m-1][j]=mat2[m-2][j];
 }
 error=0;
 for(i=0;i<m;i++)
 {
    for(j=0;j<n;j++)
    {
        error= error + pow(mat2[i][j]-mat1[i][j],2);
    }
 }
 error=sqrt(error/(m*n));

iteration++;

} while(error>0.000001);
for(i=0;i<m;i++)
{
```

```
  for(j=0;j<n;j++)
  cout<<i*dx<<"\t"<<j*dy<<"\t"<<mat2[i][j]<<endl;
}
cout<<"no. of iterations = "<<iteration;
return 0;
}
```

OUTPUT : No. of Iterations = 144



Problem_2

```
#include<iostream>
#include<math.h>
#include<stdio.h>
using namespace std;
int main()
{
  int m,n, i ,j;
  double B,dx,dy,s,w,a,expression,pi=3.1417;
  double mat1[100][100],mat2[100][100];
   cout<<"enter the size m*n\n";
  cin>>m>>n;
  dx=1.0/(m-1);
  dy=1.0/(n-1);
  B=dx/dy;
  s=pow(B,2.0);
  double l=1.0/(2.0*(1.0+s));
    expression=((cos(pi/(m-1))+s*(cos(pi/(n-1))))/(1+s));
  a=pow(expression,2);
  w=(2-sqrt(1-a))/a;
for(i=0;i<m;i++)
{
for (j = 0; j<n; j++)
  {
      if (j == (n-1))
       {
        mat2[i][j] = 0.0;
       }
      else if (i == 0)
       {
      mat2[i][j] = 1.0;
       }
      else if (j==0)
      {
      mat2[i][j] = 1.0;
      }
      else if(i== (m-1))
      {
        mat2[i][j]=1.0;
      }
      else
      {
      mat2[i][j] = 0.0;
      }
```

```
 }
}
 int iteration = 0;
 double error = 1.0;
do
{
for (i = 0; i < m; i++)
 {
 for (j = 0; j<n; j++)
  {
    mat1[i][j] = mat2[i][j];
  }
 }
  for (i = 1; i < (m-1); i++)
  {
    for (j = 1; j < (n-1); j++)
    {

     mat2[i][j]= (1-w)*mat1[i][j]+w*l*(s*mat2[i][j-1] + mat2[i-1][j]+mat1[i+1][j]
+ s*mat1[i][j+1]);
    }
  }
 error=0.0;
 for(i=0;i<m;i++)
 {
    for(j=0;j<n;j++)
    {
        error= error + pow(mat2[i][j]-mat1[i][j],2);
    }
 }
 error=sqrt(error/(m*n));

iteration++;
} while(error>0.000001);
for(i=0;i<m;i++)
{
  for(j=0;j<n;j++)
  cout<<i*dx<<"\t"<<j*dy<<"\t"<<mat2[i][j]<<endl;
}
cout<<"no. of iterations = "<<iteration;
return 0;
}
```

OUTPUT : No. of Iterations = 113