**Introduction**

**“GAME CENTER” has been designed to remove the boredom in the lives of students and give them a relief after a lot of studies. This project has been developed for fun and as a part of computer science project.**

**The game center has the following games:-**

* **Dot Matrix game**
* **Puzzle**
* **Push box**
* **Maze**
* **Tam bola**
* **Cross and knots**
* **C++ quiz**

**We have tried to create those games which are generally played manually in an attempt to make it easier for the player to play those games and to prevent any form of cheating**.

**Problems, Descriptions and Solutions**

**For example-**

1. **Tam bola which is a very popular game played in homes and at parties, has been developed on computer to prevent the quarrels that arise in it when the number of players becomes large. As the number of players increase it becomes difficult to monitor the true winner. In such a case it becomes necessary to take the help of medium like computer.**
2. **Cross and knots which has been played for a long time has been included in this project.**
3. **Dot matrix game which involves making of boxes is a very common paper pen game which is popular among students, but here the problem was making of a big grid in case of a big game. This problem is efficiently solved by this program where user can make grids according to their choice.**

**We have also included the games which have been created in an attempt to know their programming logic and learn more about the practical applications of C++ concepts.**

**For example-**

1. **Maze which has been a very popular mobile game has been included in the project.**
2. **Puzzle is a game which is also given in windows 7 as a gadget which involves solving of a 16 block puzzle has been includes in the project.**
3. **Push box which is a very common game among various platforms has been included in the program.**
4. **Snakes is a very common mobile game which is played is also included.**

**Finally we have given a C++ quiz to refresh the minds of the programmers with some basic questions involving concepts of C++ that are taught in class 12.**

**As the “Game center” keeps a track of all the players who have played the games of game center and allots space if any new entries are there so the volume of the information to be handled is therefore about millions of characters. Further the whole information has to be segmented according to the**

**Hardware and software requirements**

1. **Player name**
2. **The games which the player has played**
3. **His scores in these games**
4. **Each players’ wins and losses in the respective games**

**All this suggests that the computer should have atleast 250 MB Hard disk and about 8 to 16 MB of RAM. C++ is chosen to run the program and has been designed in it.**

**Enhancements**

**Here some of the enhancements to the program have been suggested:-**

**Bibliography**

**Books Referred:**

 Sumita Arora

 Together with C++

**Guidance through:**

 Teachers

Source Code

**#include<conio.h>**

**#include<stdio.h>**

**#include<process.h>**

**#include<stdlib.h>**

**#include<ctype.h>**

**#include<string.h>**

**#include<fstream.h>**

**#include<dos.h>**

**class menu\_pointr**

**{**

**public:**

**int menu\_pointer(int game);**

**void menu(int point);**

**};**

**void menu\_pointr :: menu(int point)**

**{**

**int s;**

**for(s=0;s<4;s++)**

**{**

**gotoxy(26,s+16);**

**cout<<" ";**

**}**

**gotoxy(26,point+16);**

**cout<<"Í";**

**}**

**int menu\_pointr :: menu\_pointer(int game)**

**{**

**float ascii,point=0;**

**char ch;**

**int s;**

**gotoxy(1,1);**

**cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";**

**for(s=0;s<5;s++)**

**{**

**if(s==2)**

**switch(game)**

**{**

**case 1: cout<<"\n \*\*\t\t\t TICK TACK GAME\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t ­­­­­­­­­­­­­­\t\t\t\t \*\*";**

**break;**

**case 2: cout<<"\n \*\*\t\t WELCOME TO THE C++ QUIZ \t\t \*\*";**

**cout<<"\n \*\*\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \t\t \*\*";**

**break;**

**case 3: cout<<"\n \*\*\t\t\t\tDOT MATRIX GAME\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t\t­­­­­­­­­­­­­­­\t\t\t\t \*\*";**

**break;**

**case 4: cout<<"\n \*\*\t\tWELCOME TO THE MAGNIFICIENT GAME OF TAMBOLA!! \t\t \*\*";**

**cout<<"\n \*\*\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \t\t \*\*";**

**break;**

**case 5: cout<<"\n \*\*\t\t\t\t MAZE WAY FIND \t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t\t ­­­­­­­­­­­­­ \t\t\t \*\*";**

**break;**

**case 6: cout<<"\n \*\*\t\t\t\tNO.PUZZULE GAME\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t\t­­­­­­­­­­­­­­­ \t\t\t \*\*";**

**break;**

**case 7: cout<<"\n \*\*\t\t\t\t PUSH BOX \t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t\t ­­­­­­­­ \t\t\t\t \*\*";**

**break;**

**case 8: cout<<"\n \*\*\t\t\t\tSNAKE GAME \t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t\t­­­­­­­­­­\t\t\t \*\*";**

**}**

**cout<<"\n \*\*\t\t\t\t\t\t\t\t\t \*\*";**

**}**

**cout<<"\n \*\*\tThis is a ";**

**switch(game)**

**{**

**case 1: cout<<"Tick Tack Game ";**

**break;**

**case 2: cout<<" C++ Quiz Game ";**

**break;**

**case 3: cout<<"Dot Matrix Game";**

**break;**

**case 4: cout<<"Game Of Tambola";**

**break;**

**case 5: cout<<"Maze yfind game";**

**break;**

**case 6: cout<<"No.Puzzule game";**

**break;**

**case 7: cout<<"Game of pushbox";**

**break;**

**case 8: cout<<"snak point game";**

**break;**

**}**

**cout<<" It is a very good game.You will like \*\*";**

**cout<<"\n \*\*\tit very much . The game is a open licence game registered by ng \*\*\n \*\*\tgames\t\t\t\t\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t\t\t\t\t\t\t \*\*\n \*\*\t\t\t\t MENU\t\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t\t ~~~~~~\t\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t 1).Start game\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t 2).Rules\t\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t 3).About the game.\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t 4).Exit\t\t\t\t\t \*\*";**

**for(s=0;s<3;s++)**

**{**

**cout<<"\n \*\*\t\t\t\t\t\t\t\t\t \*\*";**

**}**

**cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";**

**while(1)**

**{**

**menu(point);**

**while((ch=getch())==0);**

**ascii=ch;**

**if(ch=='P'||ch=='M'||ch=='s'||ch=='d')**

**point=point+1;**

**if(ch=='H'||ch=='K'||ch=='w'||ch=='a')**

**point=point-1;**

**if(point==4)**

**point=0;**

**if(point<0)**

**point=3;**

**if(ascii==13)**

**break;**

**}**

**return (point+1);**

**}**

**class dotmatrix\_game:public menu\_pointr**

**{**

**char turn,player[2][25],mat[200][200];**

**int b,ply1,ply2,x,y,escape;**

**public:**

**void outcum(char mat[200][200]);**

**int check(char mat[200][200],char turn);**

**int game();**

**void running();**

**void display(int i);**

**void mat\_pointer(int &point2,int &point,int i);**

**};**

**void dotmatrix\_game::display(int i)**

**{**

**int j;**

**gotoxy(1,1);**

**cout<<" ";**

**for(j=0;j<strlen(player[0]);j++)**

**cout<<player[0][j];cout<<"'s points = "<<ply1<<"\n";**

**cout<<" ";**

**for(j=0;j<strlen(player[1]);j++)**

**cout<<player[1][j];cout<<"'s points = "<<ply2<<"\n";**

**cout<<"\n\t\t\t DOT BOARD\n";**

**cout<<"\t\t\t ­­­­­­­­­\n\n";**

**outcum(mat);**

**if(i%2==0)**

**{**

**cout<<"\n\n\n";**

**cout<<" ";**

**for(j=0;j<strlen(player[0]);j++)**

**cout<<player[0][j];cout<<"'s turn";**

**turn=player[0][0];**

**}**

**else**

**{**

**cout<<"\n\n\n";**

**cout<<" ";**

**for(j=0;j<strlen(player[1]);j++)**

**cout<<player[1][j];cout<<"'s turn";**

**turn=player[1][0];**

**}**

**}**

**void dotmatrix\_game::mat\_pointer(int &point2,int &point,int i)**

**{**

**float ascii;**

**char ch;**

**while(1)**

**{**

**display(i);**

**while((ch=getch())==0);**

**ascii=ch;**

**if(ch=='P'||ch=='s')**

**{point=point+1;**

**if(point%2==0)**

**point2--;**

**else**

**point2++;**

**}**

**if(ch=='H'||ch=='w')**

**{point=point-1;**

**if(point%2==0)**

**point2--;**

**else**

**point2++;**

**}**

**if(ch=='M'||ch=='d')**

**point2=point2+2;**

**if(ch=='K'||ch=='a')**

**point2=point2-2;**

**if(point>b-2)**

**{point=0;**

**point2--;**

**}**

**if(point<0)**

**{point=b-2;**

**point2--;**

**}**

**if(point2>b-2&&point%2==0)**

**point2=1;**

**if(point2>b-2&&point%2!=0)**

**point2=0;**

**if(point2<0&&point%2==0)**

**point2=b-3;**

**if(point2<0&&point%2!=0)**

**point2=b-2;**

**if(ascii==13||ascii==27||ch=='e')**

**break;**

**}**

**if(ascii==27||ch=='e')**

**escape=1;**

**}**

**int dotmatrix\_game::game()**

**{**

**x=0;**

**y=1;**

**int r=0;**

**start:**

**clrscr();**

**r=menu\_pointer(3);**

**gotoxy(1,23);**

**switch(r)**

**{**

**case 2 : cout<<"\n\n The rules of the game are simple you have to make as many boxes as";**

**cout<<"\n possibe with your name the person who will put the last line of the";**

**cout<<"\n box will get his symbol in that box.To make a line you have to selct";**

**cout<<"\n position and press enter\n";**

**getch();**

**goto start;**

**case 3 : cout<<"\n\n The game is made by Nikhil garg \n It is tested by Nikhil garg. ";**

**cout<<"\n The game is made for fun and as a part of the computer project.\n The difficulty";**

**cout<<" level of";**

**cout<<" the game is null so play it and \n enjoy";**

**getch();**

**goto start;**

**case 4 : return 0;**

**case 1 :running();**

**break;**

**}**

**if(escape==1)**

**goto start;**

**cout<<"\n Thank you for playing the game!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!";**

**cout<<"\n Winning or losing is not the main thing the main thing is the that you PLAY";**

**getch();**

**return 0;**

**}**

**void dotmatrix\_game::running()**

**{**

**int i=0,j=0,alph[2],numb[2],flag=0;**

**char pt[2][2];**

**ply1=0;ply2=0;**

**escape=0;**

**clrscr();**

**cout<<"\nEnter the name of the player one = ";**

**gets(player[0]);**

**cout<<"\nEnter the name of the player two = ";**

**gets(player[1]);**

**cout<<"\nEnter the size of the cube board ";**

**cin>>b;**

**b=2\*b;**

**player[0][0]=toupper(player[0][0]);**

**player[1][0]=toupper(player[1][0]);**

**clrscr();**

**for(i=0;i<b-1;i=i+2)**

**{**

**for(j=0;j<b-1;j++,j++)**

**{**

**mat[i][j]='\*';**

**}**

**}**

**for(i=0;i<b-1;i=i+2)**

**{**

**for(j=1;j<b-1;j++,j++)**

**{**

**mat[i][j]=' ';**

**}**

**}**

**for(i=1;i<b-1;i++,i++)**

**{**

**for(j=0;j<b-1;j++)**

**{**

**mat[i][j]=' ';**

**}**

**}**

**int k=0;**

**k=((b-1)\*(b-1))/2;**

**for(i=0;i<k;i++)**

**{**

**clrscr();**

**gotoxy(53,2);**

**cout<<" Input 0 to go back to menu \n\n";**

**mat\_pointer(y,x,i);**

**if(escape==1)**

**goto end;**

**if(mat[x][y]==' ')**

**{**

**if(x%2==0)**

**mat[x][y]='-';**

**else**

**mat[x][y]='|';**

**}**

**else**

**{**

**cout<<"\nThe place is not empty ";**

**i--;**

**getch();**

**}**

**flag=check(mat,turn);**

**if(flag==1)**

**{**

**i--;**

**k--;**

**}**

**}**

**clrscr();**

**if(ply1>ply2)**

**{ cout<<"\n\t";**

**for(j=0;j<strlen(player[0]);j++)**

**cout<<player[0][j];cout<<" won by "<<ply1-ply2<<" points ";**

**}**

**else if(ply2>ply1)**

**{ cout<<"\n\t";**

**for(j=0;j<strlen(player[1]);j++)**

**cout<<player[1][j];cout<<" won by "<<ply2-ply1<<" points ";**

**}**

**else**

**cout<<"\n\tIt was a tie";**

**cout<<"\n\n";**

**end:**

**}**

**void dotmatrix\_game::outcum(char mat[200][200])**

**{**

**int i=0,j=0,k=0;**

**char a=65;**

**cout<<"\n\t\t ";**

**for(k=10;k>b/2;k--)**

**cout<<" ";**

**for(i=0;i<b/2;i++)**

**cout<<" "<<i+1;**

**cout<<"\n";**

**for(i=0;i<b-1;i++)**

**{**

**cout<<"\n\t\t";**

**for(k=10;k>b/2;k--)**

**cout<<" ";**

**for(j=0;j<b-1;j++)**

**{**

**if(i%2==0)**

**{**

**if(j==0)**

**{**

**cout<<a<<" ";**

**a++;**

**}**

**if(j%2==0)**

**cout<<mat[i][j];**

**else {**

**if(x==i&&y==j)**

**cout<<"##";**

**else**

**cout<<mat[i][j]<<mat[i][j];}**

**}**

**else**

**{**

**if(j==0)**

**cout<<" ";**

**if(j%2==0)**

**{if(i==x&&j==y)**

**cout<<"#";**

**else**

**cout<<mat[i][j];**

**}**

**else cout<<mat[i][j]<<mat[i][j];**

**}**

**}**

**}**

**}**

**int dotmatrix\_game::check(char mat[200][200],char turn)**

**{**

**int i=0,j=0,flag=0;**

**for(i=0;i<b;i++)**

**{**

**for(j=1;j<b;j++)**

**{**

**if(mat[i][j]=='-'&&mat[i+1][j-1]=='|'&&mat[i+2][j]=='-'&&mat[i+1][j+1]=='|'&&mat[i+1][j]==' ')**

**{**

**mat[i+1][j]=turn;**

**flag=1;**

**if(turn==player[0][0])**

**ply1++;**

**else if(turn==player[1][0])**

**ply2++;**

**}**

**}**

**}**

**if(flag==1)**

**return 1;**

**else**

**return 0;**

**}**

**//It is a simple quiz based on th C++ curriculum chapters which are in CBSE**

**class cquiz : public menu\_pointr**

**{**

**int i;**

**char player\_name[80];**

**char question;**

**int answer;**

**public:**

**int quiz();**

**int ques\_pointer();**

**void choice(int point);**

**};**

**void cquiz :: choice(int point)**

**{**

**gotoxy(1,3);**

**cout<<" ";**

**cout<<"\n ";**

**cout<<"\n ";**

**cout<<"\n ";**

**switch(point)**

**{**

**case 0: gotoxy(1,3);**

**cout<<"##";**

**break;**

**case 1: gotoxy(1,4);**

**cout<<"##";**

**break;**

**case 2: gotoxy(1,5);**

**cout<<"##";**

**break;**

**case 3: gotoxy(1,6);**

**cout<<"##";**

**break;**

**}**

**}**

**int cquiz :: ques\_pointer()**

**{**

**float ascii,point=0;**

**char ch;**

**while(1)**

**{**

**choice(point);**

**while((ch=getch())==0);**

**ascii=ch;**

**if(ch=='P'||ch=='s')**

**point=point+1;**

**if(ch=='H'||ch=='w')**

**point=point-1;**

**if(point==4)**

**point=0;**

**if(point<0)**

**point=3;**

**if(ascii==13)**

**break;**

**}**

**return (point+1);**

**}**

**int cquiz::quiz()**

**{**

**start:**

**clrscr();**

**i=0;**

**int s,ch;**

**ch=menu\_pointer(2);**

**//a lot of jump statements have been used to change the control the flow of**

**//program and reduce unexpected errors.**

**gotoxy(1,25);**

**switch(ch)**

**{**

**case 1:goto start2;**

**case 2:cout<<" The rules of the game are very simple. It is just an individual quiz "<<endl;**

**cout<<" test in which an individual has to answer basic C++ questions. If he "<<endl;**

**cout<<" answers all answers correctly he is declared winner else he is a loser."<<endl;**

**getch();**

**goto start;**

**case 3:cout<<" The game is made by Aditya and Nikhil association of programmers for "<<endl;**

**cout<<" people to increase their knowledge about C++ through the medium of games"<<endl;**

**getch();**

**goto start;**

**case 4:return 0;**

**};**

**start2:**

**clrscr();**

**cout<<"WELCOME TO THE C++ QUIZ"<<endl;**

**cout<<"Enter your name:";**

**gets(player\_name);**

**cout<<"Options to all the questions are (a/b/c/d)"<<endl;**

**clrscr();**

**cout<<"Q1: Constructors when declared can \_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;**

**cout<<"\n a:have a return type \n b:can't have a return type"<<endl;**

**cout<<" c:may or may not have a return type \n d:None of the above"<<endl;**

**answer=ques\_pointer();**

**gotoxy(1,8);**

**if(answer==2)**

**{**

**cout<<"The answer is correct"<<endl;**

**i++;**

**}**

**else**

**{**

**cout<<"The answer is wrong"<<endl;**

**cout<<"The correct answer is option (b)"<<endl;**

**};**

**getch();**

**clrscr();**

**cout<<"Q2: What happens when no default constructor is given by the user?"<<endl;**

**cout<<"\n a:Constructor is not created " <<endl;**

**cout<<" b:Object is created without constructor"<<endl;**

**cout<<" c:Object is created with a constructor with the help of compiler"<<endl;**

**cout<<" d:None of the above"<<endl;**

**answer=ques\_pointer();**

**gotoxy(1,8);**

**if(answer==3)**

**{**

**cout<<"The answer is correct"<<endl;**

**i++;**

**}**

**else**

**{**

**cout<<"The answer is wrong"<<endl;**

**cout<<"The correct answer is option (c)"<<endl;**

**};**

**getch();**

**clrscr();**

**cout<<"Q3: Destructors \_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;**

**cout<<"\n a:can be overloaded \n b:can't be overloaded"<<endl;**

**cout<<" c:depends on the condtion \n d:None of the above"<<endl;**

**answer=ques\_pointer();**

**gotoxy(1,8);**

**if(answer==2)**

**{**

**cout<<"The answer is correct"<<endl;**

**i++;**

**}**

**else**

**{**

**cout<<"The answer is wrong"<<endl;**

**cout<<"The correct answer is option (b)"<<endl;**

**};**

**getch();**

**clrscr();**

**cout<<"Q4: Which of the/these functions aren't inherited?"<<endl;**

**cout<<"\n a:Constructor and destructor \n b:assignment operator"<<endl;**

**cout<<" c:friend functions and friend classe \n d:All of the above"<<endl;**

**answer=ques\_pointer();**

**gotoxy(1,8);**

**if(answer==4)**

**{**

**cout<<"The answer is correct"<<endl;**

**i++;**

**}**

**else**

**{**

**cout<<"The answer is wrong"<<endl;**

**cout<<"The correct answer is option (d)"<<endl;**

**};**

**getch();**

**clrscr();**

**cout<<"Q5: Significance of colon(:) symbol is"<<endl;**

**cout<<"\n a:used after labels "<<endl;**

**cout<<" b:just an unary operator"<<endl;**

**cout<<" c:shows the relationship of derived class to base class"<<endl;**

**cout<<" d:None of the above"<<endl;**

**answer=ques\_pointer();**

**gotoxy(1,8);**

**if(answer==3)**

**{**

**cout<<"The answer is correct"<<endl;**

**i++;**

**}**

**else**

**{**

**cout<<"The answer is wrong"<<endl;**

**cout<<"The correct answer is option (c)"<<endl;**

**};**

**getch();**

**clrscr();**

**cout<<"Q6: Why cables in ethernet cable 'Twisted'?"<<endl;**

**cout<<"\n a:ethernet cables are also called as twisted pair cables"<<endl;**

**cout<<" b:To increase the transmission speed"<<endl;**

**cout<<" c:None of the above "<<endl;**

**cout<<" d:To reduce the crosstalk"<<endl;**

**answer=ques\_pointer();**

**gotoxy(1,8);**

**if(answer==4)**

**{**

**cout<<"The answer is correct"<<endl;**

**i++;**

**}**

**else**

**{**

**cout<<"The answer is wrong"<<endl;**

**cout<<"The correct answer is option (d)"<<endl;**

**};**

**getch();**

**clrscr();**

**cout<<"Q7: From where does the coaxial cable got its name?"<<endl;**

**cout<<"\n a:when it was discovered"<<endl;**

**cout<<" b:from the name of a scientist"<<endl;**

**cout<<" c:because it contains two conductors parallel to each other"<<endl;**

**cout<<" d:None of the above"<<endl;**

**answer=ques\_pointer();**

**gotoxy(1,8);**

**if(answer==3)**

**{**

**cout<<"The answer is correct"<<endl;**

**i++;**

**}**

**else**

**{**

**cout<<"The answer is wrong"<<endl;**

**cout<<"The correct answer is option (c)"<<endl;**

**};**

**getch();**

**clrscr();**

**cout<<"Q8:Popular softwares used for video conferencing is/are- "<<endl;**

**cout<<"\n a:TCP Cam \n b:Ekiga"<<endl;**

**cout<<" c:Skype \n d:All of the above"<<endl;**

**answer=ques\_pointer();**

**gotoxy(1,8);**

**if(answer==4)**

**{**

**cout<<"The answer is correct"<<endl;**

**i++;**

**}**

**else**

**{**

**cout<<"The answer is wrong"<<endl;**

**cout<<"The correct answer is option (d)"<<endl;**

**};**

**getch();**

**clrscr();**

**cout<<"Q9: IP address is a \_\_\_\_\_\_\_\_\_\_\_\_\_\_ address"<<endl;**

**cout<<"\n a:32 bit \n b:64 bit"<<endl;**

**cout<<" c:8 bit \n d:16 bit"<<endl;**

**answer=ques\_pointer();**

**gotoxy(1,8);**

**if(answer==1)**

**{**

**cout<<"The answer is correct"<<endl;**

**i++;**

**}**

**else**

**{**

**cout<<"The answer is wrong"<<endl;**

**cout<<"The correct answer is option (a)"<<endl;**

**};**

**getch();**

**clrscr();**

**cout<<"Q10: Which of these are firewall techniques"<<endl;**

**cout<<"\n a:Packet Filter \n b:Application Gateway"<<endl;**

**cout<<" c:Circuit-Level gateway \n d:All of the above"<<endl;**

**answer=ques\_pointer();**

**gotoxy(1,8);**

**if(answer==4)**

**{**

**cout<<"The answer is correct"<<endl;**

**i++;**

**}**

**else**

**{**

**cout<<"The answer is wrong"<<endl;**

**cout<<"The correct answer is option (d)"<<endl;**

**};**

**cout<<endl;**

**cout<<"Your points are:"<<i<<endl;**

**int len;**

**len=strlen(player\_name);**

**if(i==10)**

**{cout.write(player\_name,len);**

**cout<<" is a CHAMPION!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!";}**

**else**

**{cout.write(player\_name,len);**

**cout<<" is a loser!!LEARNSOMETHING!!!!LEARNSOMETHING!!!!LEARNSOMETHING!!";}**

**getch();**

**return 0;**

**}**

**class tictactoe : public menu\_pointr**

**{**

**int ply1,ply2;**

**char mat[3][3];**

**int flag,j,i,flag2,turn,t,x,y,q,escape;**

**public:**

**int check(char mat[3][3]);**

**void outcum(char mat[3][3],int x,int y);**

**int game();**

**void running();**

**void mat\_pointer(int &point2,int &point);**

**void display(int point,int point2);**

**void ques(int point2);**

**int ques\_pointer();**

**};**

**void tictactoe::display(int point,int point2)**

**{**

**gotoxy(1,1);**

**cout<<"\n Player one's pts = "<<ply1;**

**cout<<"\n Player two's pts = "<<ply2;**

**cout<<"\n\n\t\t\t\tCHECKBOARD\n";**

**cout<<"\t\t\t ­­­­­­­­­­­­­­­­­­­­­­\n\n";**

**outcum(mat,point,point2);**

**if(turn%2==0)**

**cout<<"\n\n Player one's turn";**

**else**

**cout<<"\n\n Player two's turn";**

**}**

**int tictactoe::game()**

**{**

**int r=0;**

**start:**

**clrscr();**

**r=menu\_pointer(1);**

**gotoxy(1,23);**

**switch(r)**

**{**

**case 2 : cout<<"\n\n The rules of the game are simple you have to make a line of three\n";**

**cout<<" in a row or coloum or digonal O or X. To put O or X go to the respective\n";**

**cout<<" position to enter and press enter. ";**

**getch();**

**goto start;**

**case 3 : cout<<"\n\n The game is made by Nikhil garg. It is tested by Nikhil garg. It is sold\n and registered by n.g games";**

**cout<<" The game made for fun and as a part of the \n computer project. The difficulty level of";**

**cout<<"the game is null so play it and \n have fun";**

**getch();**

**goto start;**

**case 4 : return 0;**

**case 1 : running();**

**break;**

**}**

**if(escape==1)**

**goto start;**

**return 0;**

**}**

**void tictactoe::running()**

**{**

**ply1=0,ply2=0;**

**escape=0;**

**start:**

**q='n';**

**flag=0,j=0,i=0,flag2=0,turn=0,t=0,x=0,y=0;**

**clrscr();**

**cout<<"This is a O and X game";**

**for(i=0;i<3;i++)**

**for(j=0;j<3;j++)**

**{**

**mat[i][j]=' ';**

**}**

**clrscr();**

**for(j=0;j<9;j++)**

**{**

**mat\_pointer(x,y);**

**if(escape==1)**

**goto end;**

**if(turn%2==0&&mat[y][x]==' ')**

**{**

**mat[y][x]='O';**

**}**

**else if(turn%2!=0&&mat[y][x]==' ')**

**{**

**mat[y][x]='X';**

**}**

**else**

**{**

**cout<<"\nThis location is not empty ";**

**getch();**

**j--;**

**turn--;**

**}**

**flag=check(mat);**

**if(flag==1)**

**{**

**clrscr();**

**if(turn%2==0)**

**{**

**flag2=1;**

**}**

**break;**

**}**

**else**

**clrscr();**

**turn++;**

**}**

**if(flag==1)**

**{**

**if(flag2==1)**

**{**

**cout<<"\n Player one won \n";**

**ply1++;**

**}**

**else**

**{**

**cout<<"\n Player two won \n";**

**ply2++;**

**}**

**}**

**if(flag!=1)**

**{**

**cout<<"\n No one won \n";**

**}**

**outcum(mat,4,4);**

**getch();**

**q=ques\_pointer();**

**if(q==1)**

**{**

**flag=0;**

**goto start;**

**}**

**else**

**{**

**clrscr();**

**if (ply1>ply2)**

**cout<<"\n Player one won by "<<ply1-ply2<<" points ";**

**else if(ply2>ply1)**

**cout<<"\n Player two won by "<<ply2-ply1<<" points ";**

**else**

**cout<<"\n It was a tie between player one and two. ";**

**}**

**cout<<"\n\n Thank you for playing nikhil gargs game";**

**getch();**

**end:**

**}**

**void tictactoe::outcum(char mat[3][3],int x,int y)**

**{**

**int i,j;**

**for(i=0;i<3;i++)**

**{ cout<<"\t\t\t\t" ;**

**for(j=0;j<3;j++)**

**{**

**if(x==i&&y==j)**

**cout<<'#';**

**else**

**cout<<mat[i][j];**

**if(j<2)cout<<" | ";**

**}**

**if(i<2)cout<<"\n\t\t\t\t--|---|--\n";**

**}**

**}**

**int tictactoe::check(char mat[3][3])**

**{**

**int i,j,flag=0;**

**for(i=0;i<3;i++)**

**{**

**if(mat[i][0]==mat[i][1]&&mat[i][1]==mat[i][2]&&mat[i][1]!=' ')**

**flag=1;**

**if(mat[0][i]==mat[1][i]&&mat[1][i]==mat[2][i]&&mat[1][i]!=' ')**

**flag=1;**

**}**

**if(mat[0][0]==mat[1][1]&&mat[1][1]==mat[2][2]&&mat[1][1]!=' ')**

**flag=1;**

**if(mat[0][2]==mat[1][1]&&mat[1][1]==mat[2][0]&&mat[1][1]!=' ')**

**flag=1;**

**if(flag==1)**

**return 1;**

**else**

**return 0;**

**}**

**void tictactoe::mat\_pointer(int &point2,int &point)**

**{**

**float ascii;**

**char ch;**

**while(1)**

**{**

**display(point,point2);**

**while((ch=getch())==0);**

**ascii=ch;**

**if(ch=='P'||ch=='s')**

**point=point+1;**

**if(ch=='H'||ch=='w')**

**point=point-1;**

**if(ch=='M'||ch=='d')**

**point2=point2+1;**

**if(ch=='K'||ch=='a')**

**point2=point2-1;**

**if(point==3)**

**point=0;**

**if(point<0)**

**point=2;**

**if(point2==3)**

**point2=0;**

**if(point2<0)**

**point2=2;**

**if(ascii==13||ascii==27||ch=='e')**

**break;**

**}**

**if(ascii==27||ch=='e')**

**escape=1;**

**}**

**void tictactoe :: ques(int point2)**

**{**

**int s;**

**gotoxy(9,8);**

**cout<<"\n\n\tDo you want to play again ?????\n\t";**

**switch(point2)**

**{**

**case 0:gotoxy(12,12);**

**cout<<"Yes \* \t\t\tNo ";**

**break;**

**case 1:gotoxy(12,12);**

**cout<<"Yes \t\t\tNo \*";**

**break;**

**}**

**}**

**int tictactoe :: ques\_pointer()**

**{**

**float ascii,point2=0;**

**char ch;**

**while(1)**

**{**

**ques(point2);**

**while((ch=getch())==0);**

**ascii=ch;**

**if(ch=='K'||ch=='H'||ch=='s'||ch=='a')**

**point2=point2-1;**

**if(ch=='M'||ch=='P'||ch=='w'||ch=='d')**

**point2=point2+1;**

**if(point2==2)**

**point2=0;**

**if(point2<0)**

**point2=1;**

**if(ascii==13)**

**break;**

**}**

**return (point2+1);**

**}**

**class maze : public menu\_pointr**

**{**

**char mat[3][40][60];**

**int j,i,win,randm;**

**public:**

**void outcum();**

**int game();**

**void running();**

**void mat\_pointer();**

**void display();**

**};**

**void maze::display()**

**{**

**gotoxy(1,1);**

**outcum();**

**}**

**int maze::game()**

**{**

**int r=0;**

**start:**

**clrscr();**

**r=menu\_pointer(5);**

**gotoxy(1,23);**

**switch(r)**

**{**

**case 2 : cout<<"\n\n The rules of the game are simple you have to find the way out of maze\n";**

**getch();**

**goto start;**

**case 3 : cout<<"\n\n The game is made by Nikhil garg. It is tested by Nikhil garg. It is sold\n and registered by n.g games";**

**cout<<" The game made for fun and as a part of the \n computer project. The difficulty level of";**

**cout<<"the game is null so play it and \n have fun";**

**getch();**

**goto start;**

**case 4 : return 0;**

**case 1 : running();**

**break;**

**}**

**if(win==0)**

**goto start;**

**return 0;**

**}**

**void maze::running()**

**{**

**clrscr();**

**fstream f;**

**j=0,i=0;**

**win=0,randm;**

**randm=random(3);**

**f.open("maze",ios::out|ios::in|ios::binary);**

**f.read((char\*)&mat,sizeof(mat));**

**mat\_pointer();**

**if(win==1)**

**{**

**cout<<"\n\n You have won the game \n You are a maze way finder. ";**

**cout<<"Thank you for playing nikhil gargs game";**

**getch();**

**}**

**f.close();**

**}**

**void maze::outcum()**

**{**

**int i,j;**

**for(i=0;i<40;i++)**

**{**

**cout<<"\n\t";**

**for(j=0;j<60;j++)**

**cout<<mat[randm][i][j];**

**}**

**}**

**void maze::mat\_pointer()**

**{**

**float ascii;**

**char ch;**

**int point=1,point2=1;**

**display();**

**while(1)**

**{**

**gotoxy(point2+9,point+2);**

**cout<<'';**

**gotoxy(point2+9,point+2);**

**while((ch=getch())==0);**

**ascii=ch;**

**gotoxy(point2+9,point+2);**

**cout<<' ';**

**if((ch=='P'||ch=='s')&&mat[randm][point+1][point2]!='')**

**point=point+1;**

**if((ch=='H'||ch=='w')&&mat[randm][point-1][point2]!='')**

**point=point-1;**

**if((ch=='M'||ch=='d')&&(mat[randm][point][point2+1]!=''||point2==59))**

**point2=point2+1;**

**if((ch=='K'||ch=='a')&&(mat[randm][point][point2-1]!=''||point2==0))**

**point2=point2-1;**

**if(ascii==13||mat[randm][point][point2]!=' '||ascii==27||ch=='e')**

**break;**

**}**

**if(mat[randm][point][point2]!=' ')**

**win=1;**

**}**

**class pushbox : public menu\_pointr**

**{**

**int j,i,x,y,win,randm,lvlchoice;**

**public:**

**void outcum(int x,int y);**

**int game();**

**void running();**

**void mat\_pointer(int &point2,int &point);**

**void display(int point,int point2);**

**int check();**

**void opengate(int y,int x);**

**struct pushboxdat**

**{**

**int level;**

**char mat[40][60];**

**}level;**

**};**

**void pushbox::display(int point,int point2)**

**{**

**gotoxy(1,1);**

**outcum(point,point2);**

**}**

**int pushbox::game()**

**{**

**int r=0;**

**start:**

**clrscr();**

**r=menu\_pointer(7);**

**gotoxy(1,23);**

**switch(r)**

**{**

**case 2 : cout<<"\n\n The rules of the game are simple you have to place the box in there\n";**

**cout<<" positions you can do that by draging the box. after all the boxes are \n";**

**cout<<" placed the gate will open and by passing through the gate you win.";**

**getch();**

**goto start;**

**case 3 : cout<<"\n\n The game is made by Nikhil garg. It is tested by Nikhil garg. It is sold\n and registered by n.g games";**

**cout<<" The game made for fun and as a part of the \n computer project. The difficulty level of";**

**cout<<"the game is null so play it and \n have fun";**

**getch();**

**goto start;**

**case 4 : return 0;**

**case 1 : running();**

**break;**

**}**

**if(win==0)**

**goto start;**

**return 0;**

**}**

**void pushbox::running()**

**{**

**clrscr();**

**fstream f;**

**j=0,i=0,x=1,y=1;**

**win=0;**

**f.open("pushboxlvl",ios::out|ios::in|ios::binary);**

**while(f)**

**{**

**f.read((char\*)&level,sizeof(level));**

**if(level.level==1)**

**break;**

**}**

**mat\_pointer(x,y);**

**if(win==2)**

**{**

**opengate(y,x);**

**mat\_pointer(x,y);**

**}**

**if(win==1)**

**{**

**cout<<"\n\n You have won the game \n You are a good gamer. ";**

**cout<<"Thank you for playing nikhil gargs game";**

**getch();**

**}**

**f.close();**

**}**

**void pushbox::outcum(int y,int x)**

**{**

**gotoxy(1,1);**

**int i,j;**

**for(i=0;i<40;i++)**

**{**

**cout<<"\n\t";**

**for(j=0;j<60;j++)**

**{**

**if(y==i&&x==j)**

**cout<<'';**

**else**

**cout<<level.mat[i][j];**

**}**

**}**

**}**

**void pushbox::opengate(int y,int x)**

**{**

**int i=0;**

**while(i<6)**

**{**

**level.mat[1][26-i]=' ';**

**level.mat[1][20-i]='';**

**i++;**

**sound(1700+i\*200);**

**delay(400);**

**outcum(y,x);**

**nosound();**

**}**

**}**

**void pushbox::mat\_pointer(int &point2,int &point)**

**{**

**float ascii;**

**char ch;**

**int right,left,k=0;**

**int p=0;**

**while(1)**

**{**

**if(win==0)**

**p=check();**

**if(p==2)**

**{**

**win=2;**

**break;**

**}**

**start:**

**display(point,point2);**

**while((ch=getch())==0);**

**ascii=ch;**

**if(ch=='P'||ch=='s')**

**{**

**if(level.mat[point+1][point2]!='\*'&&level.mat[point+1][point2]!='')**

**{**

**point=point+1;**

**}**

**else if(level.mat[point+1][point2]=='\*')**

**{**

**if(level.mat[point+2][point2]=='o')**

**{left=2;**

**right=2;}**

**else if(level.mat[point+2][point2-1]=='o')**

**{left=3;**

**right=0;}**

**else if(level.mat[point+2][point2+1]=='o')**

**{left=0;**

**right=3;}**

**else**

**{**

**left=0;**

**right=0;**

**}**

**k=0;**

**for(j=0;j<right;j++)**

**if(level.mat[point+4][point2+j]==' ')**

**k++;**

**for(j=0;j<left;j++)**

**if(level.mat[point+4][point2-j]==' ')**

**k++;**

**if(left==right)k--;**

**if(k==3)**

**{**

**for(j=0;j<right;j++)**

**if(level.mat[point+1][point2+j]=='\*')**

**{**

**level.mat[point+4][point2+j]='\*';**

**level.mat[point+2][point2+j]='\*';**

**level.mat[point+1][point2+j]=' ';**

**level.mat[point+3][point2+right-2]='o';**

**}**

**for(j=0;j<left;j++)**

**if(level.mat[point+1][point2-j]=='\*')**

**{**

**level.mat[point+4][point2-j]='\*';**

**level.mat[point+2][point2-j]='\*';**

**level.mat[point+3][point2-left+2]='o';**

**level.mat[point+1][point2-j]=' ';**

**}**

**point=point+1;**

**}**

**}**

**}**

**if(ch=='H'||ch=='w')**

**{**

**if(level.mat[point-1][point2]!='\*'&&level.mat[point-1][point2]!='')**

**{**

**point=point-1;**

**}**

**else if(level.mat[point-1][point2]=='\*')**

**{**

**if(level.mat[point-2][point2]=='o')**

**{left=2;**

**right=2;}**

**else if(level.mat[point-2][point2-1]=='o')**

**{left=3;**

**right=0;}**

**else if(level.mat[point-2][point2+1]=='o')**

**{left=0;**

**right=3;}**

**else**

**{**

**left=0;**

**right=0;**

**}**

**k=0;**

**for(j=0;j<right;j++)**

**if(level.mat[point-4][point2+j]==' ')**

**k++;**

**for(j=0;j<left;j++)**

**if(level.mat[point-4][point2-j]==' ')**

**k++;**

**if(left==right)k--;**

**if(k==3)**

**{**

**for(j=0;j<right;j++)**

**if(level.mat[point-1][point2+j]=='\*')**

**{**

**level.mat[point-4][point2+j]='\*';**

**level.mat[point-2][point2+j]='\*';**

**level.mat[point-3][point2+right-2]='o';**

**level.mat[point-1][point2+j]=' ';**

**}**

**for(j=0;j<left;j++)**

**if(level.mat[point-1][point2-j]=='\*')**

**{**

**level.mat[point-4][point2-j]='\*';**

**level.mat[point-2][point2-j]='\*';**

**level.mat[point-3][point2-left+2]='o';**

**level.mat[point-1][point2-j]=' ';**

**}**

**point=point-1;**

**}**

**}**

**}**

**if(ch=='M'||ch=='d')**

**{**

**if(level.mat[point][point2+1]!='\*'&&level.mat[point][point2+1]!='')**

**{**

**point2=point2+1;**

**}**

**else if(level.mat[point][point2+1]=='\*')**

**{**

**if(level.mat[point][point2+2]=='o')**

**{left=2;**

**right=2;}**

**else if(level.mat[point-1][point2+2]=='o')**

**{left=3;**

**right=0;}**

**else if(level.mat[point+1][point2+2]=='o')**

**{left=0;**

**right=3;}**

**else**

**{**

**left=0;**

**right=0;**

**}**

**k=0;**

**for(j=0;j<right;j++)**

**if(level.mat[point+j][point2+4]==' ')**

**k++;**

**for(j=0;j<left;j++)**

**if(level.mat[point-j][point2+4]==' ')**

**k++;**

**if(left==right)k--;**

**if(k==3)**

**{**

**for(j=0;j<right;j++)**

**if(level.mat[point+j][point2+1]=='\*')**

**{**

**level.mat[point+j][point2+4]='\*';**

**level.mat[point+j][point2+2]='\*';**

**level.mat[point+right-2][point2+3]='o';**

**level.mat[point+j][point2+1]=' ';**

**}**

**for(j=0;j<left;j++)**

**if(level.mat[point-j][point2+1]=='\*')**

**{**

**level.mat[point-j][point2+4]='\*';**

**level.mat[point-j][point2+2]='\*';**

**level.mat[point-left+2][point2+3]='o';**

**level.mat[point-j][point2+1]=' ';**

**}**

**point2=point2+1;**

**}**

**}**

**}**

**if(ch=='K'||ch=='a')**

**{**

**if(level.mat[point][point2-1]!='\*'&&level.mat[point][point2-1]!='')**

**{**

**point2=point2-1;**

**}**

**else if(level.mat[point][point2-1]=='\*')**

**{**

**if(level.mat[point][point2-2]=='o')**

**{left=2;**

**right=2;}**

**else if(level.mat[point-1][point2-2]=='o')**

**{left=3;**

**right=0;}**

**else if(level.mat[point+1][point2-2]=='o')**

**{left=0;**

**right=3;}**

**else**

**{**

**left=0;**

**right=0;**

**}**

**k=0;**

**for(j=0;j<right;j++)**

**if(level.mat[point+j][point2-4]==' ')**

**k++;**

**for(j=0;j<left;j++)**

**if(level.mat[point-j][point2-4]==' ')**

**k++;**

**if(left==right)k--;**

**if(k==3)**

**{**

**for(j=0;j<right;j++)**

**if(level.mat[point+j][point2-1]=='\*')**

**{**

**level.mat[point+j][point2-4]='\*';**

**level.mat[point+j][point2-2]='\*';**

**level.mat[point+right-2][point2-3]='o';**

**level.mat[point+j][point2-1]=' ';**

**}**

**for(j=0;j<left;j++)**

**if(level.mat[point-j][point2-1]=='\*')**

**{**

**level.mat[point-j][point2-4]='\*';**

**level.mat[point-j][point2-2]='\*';**

**level.mat[point-left+2][point2-3]='o';**

**level.mat[point-j][point2-1]=' ';**

**}**

**point2=point2-1;**

**}**

**}**

**}**

**if(ascii==13||level.mat[point][point2]!=' '||ascii==27||ch=='e')**

**break;**

**}**

**if(level.mat[point][point2]!=' ')**

**win=1;**

**}**

**int pushbox::check()**

**{**

**int flag=0;**

**for(i=0;i<40;i++)**

**for(j=0;j<60;j++)**

**if(level.mat[i][j]=='\*'&&level.mat[i+1][j+2]=='o'&&level.mat[i+3][j+3]=='\*')**

**{**

**flag++;**

**if(flag==2)**

**break;**

**}**

**if(flag==2)**

**return 2;**

**else**

**return 0;**

**}**

**class puzzule : public menu\_pointr**

**{**

**int mat[5][4];**

**int x,y,win,randm,a[19];**

**public:**

**void outcum();**

**void initialize\_mat();**

**int random\_no\_generate();**

**int game();**

**void running();**

**void mat\_pointer();**

**void display();**

**int check();**

**};**

**void puzzule::initialize\_mat()**

**{**

**int i,j;**

**for(i=0;i<20;i++)**

**{**

**a[i]=i;**

**}**

**for(i=0;i<5;i++)**

**for(j=0;j<4;j++)**

**mat[i][j]=random\_no\_generate();**

**}**

**int puzzule::random\_no\_generate()**

**{**

**int temp;**

**start:**

**randm=random(20);**

**if(a[randm]==-1)**

**goto start;**

**temp=a[randm];**

**a[randm]=-1;**

**return temp;**

**}**

**void puzzule::display()**

**{**

**gotoxy(1,1);**

**outcum();**

**}**

**int puzzule::game()**

**{**

**int r=0;**

**start:**

**clrscr();**

**r=menu\_pointer(6);**

**gotoxy(1,23);**

**switch(r)**

**{**

**case 2 : cout<<"\n\n The rules of the game are simple you have to arrange all the no in \n";**

**cout<<" accending order and you can do that by changing boxes positions only boxes \n";**

**cout<<" on the side of blank cell can be moved. once all boxes are arranged you win. ";**

**getch();**

**goto start;**

**case 3 : cout<<"\n\n The game is made by Nikhil garg. It is tested by Nikhil garg. It is sold\n and registered by n.g games";**

**cout<<" The game made for fun and as a part of the \n computer project. The difficulty level of";**

**cout<<"the game is null so play it and \n have fun";**

**getch();**

**goto start;**

**case 4 : return 0;**

**case 1 : running();**

**break;**

**}**

**if(win==0)**

**goto start;**

**return 0;**

**}**

**void puzzule::running()**

**{**

**clrscr();**

**win=0;**

**x=0,y=0;**

**initialize\_mat();**

**mat\_pointer();**

**display();**

**if(win==1)**

**{**

**cout<<"\n\n\n You have won the game \n You are great puzzule solver";**

**cout<<" Thank you for playing nikhil gargs game";**

**getch();**

**}**

**}**

**void puzzule::outcum()**

**{**

**int i,j;**

**cout<<"\n\t";**

**for(i=0;i<30;i++)**

**cout<<'';**

**for(i=0;i<5;i++)**

**{**

**cout<<"\n\t";**

**for(j=0;j<4;j++)**

**{**

**if(y==i&&x==j)**

**cout<<" ";**

**else**

**cout<<"\*-----\*";**

**}**

**cout<<"\n\t";**

**for(j=0;j<4;j++)**

**{**

**if(y==i&&x==j)**

**cout<<" ";**

**else**

**cout<<"| |";**

**}**

**cout<<"\n\t";**

**for(j=0;j<4;j++)**

**{**

**if(y==i&&x==j)**

**cout<<" ";**

**else**

**{**

**if(mat[i][j]<10)**

**cout<<"| "<<mat[i][j]<<" |";**

**else**

**cout<<"| "<<mat[i][j]<<" |";**

**}**

**}**

**cout<<"\n\t";**

**for(j=0;j<4;j++)**

**{**

**if(y==i&&x==j)**

**cout<<" ";**

**else**

**cout<<"| |";**

**}**

**cout<<"\n\t";**

**for(j=0;j<4;j++)**

**{**

**if(y==i&&x==j)**

**cout<<" ";**

**else**

**cout<<"\*-----\*";**

**}**

**}**

**cout<<"\n\t";**

**for(i=0;i<30;i++)**

**cout<<'';**

**for(i=0;i<25;i++)**

**{**

**gotoxy(38,3+i);**

**cout<<"";**

**}**

**}**

**void puzzule::mat\_pointer()**

**{**

**float ascii;**

**char ch;**

**int i,j,p;**

**for(i=0;i<5;i++)**

**for(j=0;j<4;j++)**

**if(mat[i][j]==0)**

**{**

**y=i;**

**x=j;**

**break;**

**}**

**while(1)**

**{**

**p=check();**

**if(p)**

**break;**

**display();**

**while((ch=getch())==0);**

**ascii=ch;**

**if((ch=='P'||ch=='s')&&y!=0)**

**{**

**mat[y][x]=mat[y-1][x];**

**mat[y-1][x]=0;**

**y=y-1;**

**}**

**if((ch=='H'||ch=='w')&&y!=4)**

**{**

**mat[y][x]=mat[y+1][x];**

**mat[y+1][x]=0;**

**y=y+1;**

**}**

**if((ch=='M'||ch=='d')&&x!=0)**

**{**

**mat[y][x]=mat[y][x-1];**

**mat[y][x-1]=0;**

**x=x-1;**

**}**

**if((ch=='K'||ch=='a')&&x!=3)**

**{**

**mat[y][x]=mat[y][x+1];**

**mat[y][x+1]=0;**

**x=x+1;**

**}**

**if(ascii==13||ascii==27||ch=='e')**

**break;**

**}**

**if(p==1)**

**win=1;**

**}**

**int puzzule::check()**

**{**

**int i=0,j=0,k=0,flag=0;**

**for(i=0;i<5;i++)**

**{**

**for(j=0;j<4;j++)**

**{**

**if(mat[i][j]==0)**

**continue;**

**k++;**

**if(mat[i][j]!=k)**

**{**

**flag=0;**

**break;**

**}**

**else**

**flag=1;**

**}**

**if(flag==0)**

**break;**

**}**

**return flag;**

**}**

**struct snake\_location**

**{**

**int cord\_x;**

**int cord\_y;**

**snake\_location \*next;**

**snake\_location \*previous;**

**}\*front,\*rear;**

**//controls the snake size and movement**

**class snake : public menu\_pointr**

**{**

**char mat[30][60];//grid is stored**

**int j,i,win,randm,length,ans;**

**public:**

**snake()**

**{**

**front->cord\_x=15;**

**front->cord\_y=15;**

**front->next=rear;**

**front->previous=NULL;**

**rear->cord\_x=16;**

**rear->cord\_y=15;**

**rear->next=NULL;**

**rear->previous=front;**

**length=0;**

**}//initializes**

**void outcum();//**

**int game();**

**void running();**

**void mat\_pointer();**

**void display();**

**~snake();//deletes the linked list**

**void snake\_move(int start\_x,int start\_y,int point);**

**void random\_no\_generate();//point position where snake has to go**

**int snake\_check(int start\_x,int start\_y);//checks whther player out or not**

**void ques(int point2);//do you want to start prev game**

**int ques\_pointer();//moves pointer for yes and know**

**};**

**void snake::random\_no\_generate()**

**{**

**int r\_cord\_y,r\_cord\_x;**

**start:**

**r\_cord\_y=(rand()%28)+1;**

**r\_cord\_x=(rand()%58)+1;**

**snake\_location \*temp;**

**temp=front;**

**while(temp!=NULL)**

**{**

**if(temp->cord\_x==r\_cord\_x&&temp->cord\_y==r\_cord\_y)**

**goto start;**

**temp=temp->next;**

**}**

**if(mat[r\_cord\_y][r\_cord\_x]=='\*')**

**goto start;**

**mat[r\_cord\_y][r\_cord\_x]='';**

**gotoxy(r\_cord\_x+9,r\_cord\_y+2);**

**cout<<'';**

**}**

**void snake::snake\_move(int start\_x,int start\_y,int point)**

**{**

**if(point==0)**

**{**

**snake\_location \*temp;**

**temp=rear;**

**rear=rear->previous;**

**rear->next=NULL;**

**gotoxy(temp->cord\_x+9,temp->cord\_y+2);**

**cout<<' ';**

**delete temp;**

**length--;**

**}**

**gotoxy(front->cord\_x+9,front->cord\_y+2);**

**cout<<'';**

**snake\_location \*temp2=new snake\_location;**

**temp2->next=front;**

**temp2->previous=NULL;**

**front->previous=temp2;**

**front=temp2;**

**front->cord\_x=start\_x;**

**front->cord\_y=start\_y;**

**gotoxy(front->cord\_x+9,front->cord\_y+2);**

**cout<<'';**

**length++;**

**delay(100);**

**gotoxy(10,34);**

**cout<<"SCORE="<<(10\*length);**

**}**

**snake::~snake()**

**{**

**snake\_location \*temp;**

**while(front!=NULL)**

**{**

**temp=front;**

**front=front->next;**

**delete temp;**

**}**

**}**

**void snake::display()**

**{**

**gotoxy(1,1);**

**outcum();**

**}**

**int snake::game()**

**{**

**int r=0;**

**ans=0,no\_game=-1;**

**start:**

**no\_game++;**

**clrscr();**

**r=menu\_pointer(8);**

**gotoxy(1,23);**

**switch(r)**

**{**

**case 2 : cout<<"\n\n The rules of the game are simple you have to find the way out of maze\n";**

**getch();**

**goto start;**

**case 3 : cout<<"\n\n The game is made by Nikhil garg. It is tested by Nikhil garg. It is sold\n and registered by n.g games";**

**cout<<" The game made for fun and as a part of the \n computer project. The difficulty level of";**

**cout<<"the game is null so play it and \n have fun";**

**getch();**

**goto start;**

**case 4 : return 0;**

**case 1 :running();**

**break;**

**}**

**if(win==0)**

**goto start;**

**return 0;**

**}**

**void snake::ques(int point2)**

**{**

**switch(point2)**

**{**

**case 0:**

**gotoxy(30,27);**

**cout<<' ';**

**gotoxy(20,27);**

**cout<<'\*';**

**break;**

**case 1:**

**gotoxy(20,27);**

**cout<<' ';**

**gotoxy(30,27);**

**cout<<'\*';**

**break;**

**}**

**}**

**int snake::ques\_pointer()**

**{**

**float ascii,point2=0;**

**char ch;**

**gotoxy(10,25);**

**cout<<"Do you want restart previous game ?????";**

**gotoxy(10,27);**

**cout<<" Yes No";**

**while(1)**

**{**

**ques(point2);**

**while((ch=getch())==0);**

**ascii=ch;**

**if(ch=='K'||ch=='H'||ch=='w'||ch=='a')**

**point2=point2-1;**

**if(ch=='M'||ch=='P'||ch=='s'||ch=='d')**

**point2=point2+1;**

**if(point2==2)**

**point2=0;**

**if(point2<0)**

**point2=1;**

**if(ascii==13)**

**break;**

**}**

**return (point2);**

**}**

**void snake::running()**

**{**

**fstream f;**

**j=0,i=0;**

**win=0;**

**if(no\_game)**

**ans=ques\_pointer();**

**if(ans)**

**{**

**front->cord\_x=15;**

**front->cord\_y=15;**

**front->next=rear;**

**front->previous=NULL;**

**rear->cord\_x=16;**

**rear->cord\_y=15;**

**rear->next=NULL;**

**rear->previous=front;**

**length=0;**

**}**

**clrscr();**

**f.open("snake",ios::out|ios::in|ios::binary);**

**f.read((char\*)&mat,sizeof(mat));**

**mat\_pointer();**

**if(win==1)**

**{**

**cout<<"\n\n You have won the game \n You are a maze way finder. ";**

**cout<<"Thank you for playing nikhil gargs game";**

**getch();**

**}**

**if(ans)**

**{**

**snake\_location \*temp;**

**while(front!=NULL)**

**{**

**temp=front;**

**front=front->next;**

**delete temp;**

**}**

**}**

**f.close();**

**}**

**void snake::outcum()**

**{**

**int i,j;**

**for(i=0;i<30;i++)**

**{**

**cout<<"\n\t";**

**for(j=0;j<60;j++)**

**cout<<mat[i][j];**

**}**

**}**

**void snake::mat\_pointer()**

**{**

**float ascii;**

**char ch,oldch;**

**int i=-1,start\_y=front->cord\_y,start\_x=front->cord\_x,point=0;**

**display();**

**snake\_move(start\_x,start\_y,point);**

**getch();**

**while(1)**

**{**

**snake\_move(start\_x,start\_y,point);**

**point=0;**

**if(kbhit())**

**{**

**while((ch=getch())==0);**

**if(ch=='K'&&oldch=='M'||ch=='M'&&oldch=='K'||ch=='P'&&oldch=='H'||ch=='H'&&oldch=='P')**

**ch=oldch;**

**ascii=ch;**

**oldch=ch;**

**}**

**if(ch=='P')**

**start\_y=start\_y+1;**

**if(ch=='H')**

**start\_y=start\_y-1;**

**if(ch=='M')**

**start\_x=start\_x+1;**

**if(ch=='K')**

**start\_x=start\_x-1;**

**if(start\_y==0)**

**start\_y=28;**

**if(start\_y==29)**

**start\_y=1;**

**if(start\_x==0)**

**start\_x=58;**

**if(start\_x==59)**

**start\_x=1;**

**if(mat[start\_y][start\_x]=='')**

**{mat[start\_y][start\_x]=' ';**

**point=1;**

**random\_no\_generate();**

**}**

**if((snake\_check(start\_x,start\_y)&&i>length)||mat[start\_y][start\_x]=='\*')**

**{**

**gotoxy(30,36);**

**cout<<"OUT!!!!!!!!!";**

**snake\_move(start\_x,start\_y,point);**

**getch();**

**break;**

**}**

**if(ascii==27)**

**break;**

**if(ascii==13)**

**{**

**gotoxy(30,36);**

**cout<<"PAUSED";**

**while((ch=getch())==13);**

**gotoxy(30,36);**

**cout<<" ";**

**}**

**i++;**

**}**

**}**

**int snake::snake\_check(int start\_x,int start\_y)**

**{**

**snake\_location \*temp;**

**for(temp=front->next;temp!=NULL;temp=temp->next)**

**{**

**if(temp->cord\_x==start\_x&&temp->cord\_y==start\_y)**

**return 1;**

**}**

**return 0;**

**}**

**class tambola : public menu\_pointr**

**{**

**static int chance;//this has been used to count the number of chances taken**

**//to complete the game b/w the players.**

**int c[400],players,i,j,k,a[100][10][10],num[100];**

**char player\_name[100][80];**

**public:**

**void assign();**

**void competition();**

**void tickets(int t);**

**int play();**

**void announcer\_display();**

**void announcer();**

**int realtime\_play();**

**void tikets\_of\_players(int t);**

**};**

**int tambola::chance=0;**

**void tambola::assign()**

**{**

**//this is used for assigning the values to the array which is used later for**

**//storing temp values and checking whether a number is repeated or not.**

**for(i=0;i<400;i++)**

**{**

**c[i]=0;**

**};**

**};**

**void tambola::competition()**

**{**

**//Actually in this game only the main game proceedes because it does most of**

**//the checking work and calls various functions for the required purpose.**

**//Apart from that it also accesses player ticket and name and modifies acc.**

**//to the rules of the game and keeps the game going on.**

**int f=0,p,flag=0,gamer=0;**

**do**

**{**

**clrscr();**

**gamer=realtime\_play();//realtime play gives a non repeated random value(1-99)**

**cout<<"\t\tThe announced number is"<<gamer<<endl;**

**for(p=0;p<players;p++)**

**{**

**cout<<endl;**

**cout<<"\tTicket of the player:";**

**puts(player\_name[p]);**

**cout<<"\tis -->"<<endl;**

**tikets\_of\_players(p);**

**//loop for checking whether announced number is in ticket or not.**

**for(i=0;i<3;i++)**

**{**

**for(j=0;j<11;j++)**

**{**

**if( (i==0&&(j==0||j==2||j==4||j==6||j==8))||(i==2&&(j==0||j==2||j==4||j==6||j==8))||(i==1&&(j==1||j==3||j==5||j==7)))**

**{**

**if(a[p][i][j]==gamer)**

**{a[p][i][j]=0;}**

**}**

**}**

**};**

**getch();**

**};**

**int check=0;**

**//loop for checking the winner of the game**

**for(p=0;p<players;p++)**

**{**

**check=0;**

**for(i=0;i<3;i++)**

**{**

**for(j=0;j<11;j++)**

**{**

**if( (i==0&&(j==0||j==2||j==4||j==6||j==8))||(i==2&&(j==0||j==2||j==4||j==6||j==8))||(i==1&&(j==1||j==3||j==5||j==7)))**

**{**

**if(a[p][i][j]==0)**

**check++;**

**if(check==14)**

**{**

**cout<<"Winner of the game is"<<endl;**

**puts(player\_name[p]);**

**f=1;**

**}**

**}**

**}**

**}**

**};**

**chance++;**

**cout<<"The chance is:"<<chance<<endl;**

**}while(f!=1);**

**};//end of function**

**void tambola::tikets\_of\_players(int t=0)**

**{**

**//This function is responsible for displaying the tickets.**

**for(i=0;i<3;i++)**

**{**

**cout<<"\t\t";**

**for(j=0;j<11;j++)**

**{**

**if( (i==0&&(j==0||j==2||j==4||j==6||j==8))||(i==2&&(j==0||j==2||j==4||j==6||j==8))||(i==1&&(j==1||j==3||j==5||j==7)))**

**{if(a[t][i][j]<10)**

**cout<<" "<<a[t][i][j];**

**else**

**cout<<a[t][i][j];}**

**else**

**{cout<<" ";}**

**}**

**cout<<endl;**

**};**

**};**

**void tambola::announcer\_display()**

**{**

**//this function displays the grid of numbers in which the game is played.**

**k=1;**

**for(i=0;i<9;i++)**

**{**

**cout<<" ";**

**for(j=0;j<11;j++)**

**{**

**if(num[k]<10)**

**cout<<num[k]<<" ";**

**else**

**cout<<num[k]<<" ";**

**k++;**

**}**

**cout<<endl<<endl;**

**};**

**};**

**int tambola::realtime\_play()**

**{**

**//this function evaluates carefully whether a number has been previously**

**//announced or not and then returns a value to be announced.**

**int temp;**

**i=0;**

**start:**

**i=0;**

**temp=random(99)+1;**

**while(c[i]!=0)**

**{**

**if(c[i]==temp)**

**goto start;**

**i++;**

**};**

**c[i]=temp;**

**i=0;**

**return (temp);**

**};**

**void tambola::announcer()**

**{**

**//it is responsible for assigning values.**

**for(i=1;i<=99;i++)**

**num[i]=i;**

**assign();**

**};**

**int tambola::play()**

**{**

**//void main();**

**//the function which contains the welcome screen and which is responsible**

**//for starting the game of tambola and gives option from exiting as well.**

**start:**

**clrscr();**

**int ch,s,flag=0,tic=0,player=0,i=0,j=0;**

**ch=menu\_pointer(4);**

**//a lot of jump statements have been used to change the control the flow of**

**//program and reduce unexpected errors.**

**gotoxy(1,25);**

**switch(ch)**

**{**

**case 1:goto start2;**

**break;**

**case 2:cout<<" The rules of the game are very simple. The game is played in b/w the "<<endl;**

**cout<<" the number of players you specify. In this game you just have to check "<<endl;**

**cout<<" which of the announced number matches your ticket number. The computer "<<endl;**

**cout<<" announces the number randomly and if any number matches your ticket it "<<endl;**

**cout<<" turns to 0 in the next turn automatically. The winner is the player "<<endl;**

**cout<<" whose all numbers first become zero. Here you may be lucky if you have "<<endl;**

**cout<<" same number multiple times on your ticket. If more than one player wins"<<endl;**

**cout<<" all those players are the winners of the game which means there can be "<<endl;**

**cout<<" a tie. "<<endl;**

**getch();**

**goto start;**

**break;**

**case 3:cout<<" The game is made by Aditya and Nikhil association of programmers for "<<endl;**

**cout<<" people to enjoy the traditional games like tambola on the computer. "<<endl;**

**getch();**

**goto start;**

**break;**

**case 4:return 0;**

**};**

**start2:**

**clrscr();**

**//the game starts from here.**

**cout<<"The list of the numbers in which the tambola is going to be played is:-"<<endl;**

**announcer();//does the initialization**

**announcer\_display();//displays the grid of numbers.**

**cout<<"Enter the no. of the players"<<endl;**

**cin>>players;**

**tickets(players);//generates the ticket for players.**

**for(i=0;i<players;i++)**

**{**

**cout<<"Enter the name of the player "<<i+1<<":";**

**gets(player\_name[i]);**

**}**

**//loop for giving player names and alloting tickets.**

**for(tic=0;tic<players;tic++)**

**{**

**cout<<"The name of the player "<<tic+1<<":";**

**puts(player\_name[tic]);**

**cout<<"The ";**

**cout<<" ticket for the above player is:-"<<endl<<endl;**

**for(i=0;i<3;i++)**

**{**

**cout<<"\t\t";**

**for(j=0;j<11;j++)**

**{**

**if( (i==0&&(j==0||j==2||j==4||j==6||j==8))||(i==2&&(j==0||j==2||j==4||j==6||j==8))||(i==1&&(j==1||j==3||j==5||j==7)))**

**{**

**if(a[tic][i][j]<10)**

**cout<<" "<<a[tic][i][j];**

**else**

**cout<<a[tic][i][j];**

**}**

**else**

**{**

**cout<<" ";**

**}**

**}**

**cout<<endl;**

**}**

**cout<<endl;**

**};**

**getch();**

**competition();**

**getch();**

**chance=0;**

**start3:**

**};**

**void tambola::tickets(int t)**

**{**

**/\*this function is used to make the number of tickets required\*/**

**randomize();**

**for(k=0;k<t;k++)**

**{**

**for(i=0;i<3;i++)**

**{**

**for(j=0;j<11;j++)**

**{**

**if( (i==0&&(j==0||j==2||j==4||j==6||j==8))||(i==2&&(j==0||j==2||j==4||j==6||j==8))||(i==1&&(j==1||j==3||j==5||j==7)) )**

**{a[k][i][j]=random(99)+1;}**

**else**

**{a[k][i][j]=1; }**

**}**

**}**

**};**

**};**

**void menu(int point)**

**{**

**int s;**

**for(s=0;s<9;s++)**

**{**

**gotoxy(18,s+15);**

**cout<<" ";**

**}**

**gotoxy(18,point+15);**

**cout<<"Í";**

**}**

**int menu\_pointer()**

**{**

**float ascii,point=0;**

**char ch;**

**int s;**

**randomize();**

**gotoxy(1,1);**

**cout<<"\n \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_";**

**cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";**

**for(s=0;s<4;s++)**

**{**

**if(s==2)**

**{**

**cout<<"\n \*\*\t\t\tWELCOME TO OUR GAME CENTER \t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~ \t\t\t \*\*";**

**}**

**if(s==2)**

**cout<<"\n \*\*\t\t\t\t\t\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t\t\t\t\t\t\t \*\*";**

**}**

**cout<<"\n \*\*\t\t\t\t\t\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t\t\t\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\tEnter the game you want to play\t\t\t \*\*";**

**cout<<"\n \*\*\t\t\t\t\t\t\t\t\t \*\*";**

**cout<<"\n \*\*\t\t 1. Cross and knots \t\t\t \*\*";**

**cout<<"\n \*\*\t\t 2. C++ Quiz \t\t\t \*\*";**

**cout<<"\n \*\*\t\t 3. DOT Matrix game \t\t\t \*\*";**

**cout<<"\n \*\*\t\t 4. Tambola \t\t\t \*\*";**

**cout<<"\n \*\*\t\t 5. Maze Yfind game \t\t\t \*\*";**

**cout<<"\n \*\*\t\t 6. No.puzzule game \t\t\t \*\*";**

**cout<<"\n \*\*\t\t 7. Push box \t\t\t \*\*";**

**cout<<"\n \*\*\t\t 5. Snake game \t\t\t \*\*";**

**cout<<"\n \*\*\t\t 8. Exit from the center \t\t\t \*\*";**

**for(s=0;s<3;s++)**

**{**

**cout<<"\n \*\*\t\t\t\t\t\t\t\t\t \*\*";**

**}**

**cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";**

**while(1)**

**{**

**menu(point);**

**while((ch=getch())==0);**

**ascii=ch;**

**if(ch=='P'||ch=='M'||ch=='s'||ch=='d')**

**point=point+1;**

**if(ch=='H'||ch=='K'||ch=='w'||ch=='a')**

**point=point-1;**

**if(point==9)**

**point=0;**

**if(point<0)**

**point=8;**

**if(ascii==13)**

**break;**

**}**

**return (point+1);**

**}**

**void ques(int point2)**

**{**

**int s;**

**clrscr();**

**cout<<"\n\n\tDo you want to come again to the game center?????\n\t";**

**switch(point2)**

**{**

**case 0:cout<<"\n\t\tYes \* \t\t\tNo";**

**break;**

**case 1:cout<<"\n\t\tYes \t\t\tNo \*";**

**break;**

**}**

**}**

**int ques\_pointer()**

**{**

**float ascii,point2=0;**

**char ch;**

**while(1)**

**{**

**ques(point2);**

**while((ch=getch())==0);**

**ascii=ch;**

**if(ch=='K'||ch=='H'||ch=='w'||ch=='a')**

**point2=point2-1;**

**if(ch=='M'||ch=='P'||ch=='s'||ch=='d')**

**point2=point2+1;**

**if(point2==2)**

**point2=0;**

**if(point2<0)**

**point2=1;**

**if(ascii==13)**

**break;**

**}**

**return (point2+1);**

**}**

**int main()**

**{**

**dotmatrix\_game s5;**

**cquiz s2;**

**tambola s3;**

**tictactoe s8;**

**maze s10;**

**puzzule s12;**

**pushbox s14;**

**snake s16;**

**int ch=0,i;**

**int choice=1;**

**do**

**{**

**clrscr();**

**ch=menu\_pointer();;**

**switch(ch)**

**{**

**case 1:s8.game();**

**break;**

**case 2:s2.quiz();**

**break;**

**case 3:s5.game();**

**break;**

**case 4:s3.play();**

**break;**

**case 5:s10.game();**

**break;**

**case 6:s12.game();**

**break;**

**case 7:s14.game();**

**break;**

**case 8:s16.game();**

**break;**

**case 9:goto end;**

**}**

**choice=ques\_pointer();**

**}while(choice==1);**

**end:**

**clrscr();**

**cout<<endl<<endl;**

**cout<<"\tThanks a lot for visiting our game center and playing our games"<<endl<<endl;**

**cout<<"\tBest wishes for life from Nikhil and Aditya!!!!!!!!!!!!!!!!!!!!"<<endl;**

**getch();**

**return 0;**

**}**

**Header Files Used**

|  |  |
| --- | --- |
| Header Files | Functions Used |
| fstream.h | File based functions:  • open()  • close()  • seekg()  • seekp()  • tellg()  • read()  • write()  • cin  • cout |
| stdio.h | • gets()  • puts() |
| conio.h | • getch()  • clrscr() |
| string.h | • strcmp() |
| process.h | • exit() |
| cytpe.h | * isupper() * islower() * toupper() * tolower() |
| stdlib.h | * randomize() |
| dos.h | * sound * nosound |

**Classes and Structures used**

**Classes used:-**

1. **menu\_pointer**

|  |  |  |
| --- | --- | --- |
| Member functions | | |
| Accessibility | **Name** | **Details** |
| public | **int menu\_pointer(int game);** | **Evaluates which menu to be displayed** |
| **void menu(int point);** | **It displays the point ’=>’.** |

1. **dot-matrix\_game:public menu\_pointr**

|  |  |  |
| --- | --- | --- |
| Data members | | |
| Accessibility | **Name** | **Details** |
| Private | **char turn,player[2][25],mat[200][200];** | **mat[200][200]-for matrix**  **player[2][25]-names of players**  **turn-for giving player name in the box** |
| **int b,ply1,ply2,x,y,escape;** | **ply1,ply 2-points**  **escape-checks either win or esc or enter to exit** |
| Member functions | | |
| Accessibility | **Name** | **Details** |
| Public | **void outcum(char mat[200][200]);** | **prints the matrix** |
| **int check(char mat[200][200],char turn);** | **checks whether the box is created or not** |
| **int game();** | **this function helps in choosing the options** |
| **void running();** | **the game runs in this function.** |
| **void display(int i);** | **the overall output ids displayed by this function** |
| **void mat\_pointer(int &point2,int &point,int i);** | **it displays the pointer in the matrix in which the game is played.** |

1. **cquiz : public menu\_pointr**

|  |  |  |
| --- | --- | --- |
| Data members | | |
| Accessibility | **Name** | **Details** |
| private | **int i;** | **for loop** |
| **char player\_name[80];** | **to store the name of the player** |
| **char question;** | **Extra variables in case required.** |
| **int answer;** |
| Member functions | | |
| Accessibility | **Name** | **Details** |
| public | **int quiz();** | **it is the function in which the quiz is played.** |
| **int ques\_pointer();** | **displays the question** |
| **void choice(int point);** | **displays and evaluates the pointer** |

**4) tictactoe : public menu\_pointr**

|  |  |  |
| --- | --- | --- |
| Data members | | |
| Accessibility | **Name** | **Details** |
| private | **int ply1,ply2;** | **points of player 1 & 2** |
| **char mat[3][3];** | **matrix of the tic tac toe** |
| **int flag,j,i,flag2,turn,t,x,y,q,escape;** | **these variables are just used for loops and checking.** |
| Member functions | | |
| Accessibility | **Name** | **Details** |
| public | **int check(char mat[3][3]);** | **checks whether any one has won** |
| **void outcum(char mat[3][3],int x,int y);** | **prints the tic tac toe** |
| **int game();** | **this function helps in choosing the options** |
| **void running();** | **the game runs in this function.** |
| **void mat\_pointer(int &point2,int &point);** | **it displays the pointer in the matrix in which the game is played.** |
| **void display(int point,int point2);** | **the overall output ids displayed by this function** |
| **void ques(int point2);** | **displays and evaluates the pointer** |
| **int ques\_pointer();** | **displays the question** |

**5) maze : public menu\_pointr**

|  |  |  |
| --- | --- | --- |
| Data members | | |
| Accessibility | **Name** | **Details** |
| Private | **char mat[3][40][60];** | **stores the maze** |
| **int j,i,win,randm;** | **i,j-loops**  **win-checks winning**  **randm-choses the maze from the set of three matrix** |
| Member functions | | |
| Accessibility | **Name** | **Details** |
| public | **void outcum();** | **prints the maze** |
| **int game();** | **this function helps in choosing the options** |
| **void running();** | **the game runs in this function.** |
| **void mat\_pointer();** | **it displays the pointer in the maze in which the game is played.** |
| **void display();** | **the overall output ids displayed by this function** |

**6) pushbox : public menu\_pointr**

|  |  |  |
| --- | --- | --- |
| Data members | | |
| Accessibility | **Name** | **Details** |
| private | **int j,i,x,y,win,randm,lvlchoice;** | **i,j-loops**  **x,y-coordinates of pointers**  **win-checks winning**  **randm-choses the maze from the set of three matrix** |
| Member functions | | |
| Accessibility | **Name** | **Details** |
| public | **void outcum(int x,int y);** | **prints the pushbox** |
| **int game();** | **this function helps in choosing the options** |
| **void running();** | **the game runs in this function.** |
| **void mat\_pointer(int &point2,int &point);** | **it displays the pointer in the maze in which the game is played.** |
| **void display(int point,int point2);** | **the overall output ids displayed by this function** |
| **int check();** | **checks whether any one has won** |
| **struct pushboxdat**  **{**  **int level,winpos[3];**  **char mat[40][60];**  **}level;** | **for levels.** |

**7) puzzule: public menu\_pointr**

|  |  |  |
| --- | --- | --- |
| Data members | | |
| Accessibility | **Name** | **Details** |
| private | **int mat[5][4];** | **displays the puzzle** |
| **int x,y,win,randm,a[19];** | **i,j-loops**  **x,y-coordinates of pointers**  **win-checks winning**  **randm-choses the maze from the set of three matrix** |
| Member functions | | |
| Accessibility | **Name** | **Details** |
| public | **void outcum();** | **prints the pushbox** |
| **void initialize\_mat();** | **numbers are feeded by this function.** |
| **int random\_no\_generate();** | **generates the random number** |
| **int game();** | **this function helps in choosing the options** |
| **void running();** | **the game runs in this function.** |
| **void mat\_pointer();** | **it displays the pointer in the maze in which the game is played.** |
| **void display();** | **the overall output ids displayed by this function** |
| **int check();** | **checks whether any one has won** |

**8) tambola : public menu\_pointr**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data members | | | | |
| Accessibility | **Name** | | **Details** | |
| private | **static int chance;** | | **It counts the number of chances in which the game gets over.** | |
| **int c[400],players, i , j, k,a[100][10][10],num[100];** | | **c[400] stores the random value,i,j,k, are used in loops, a[100][10][10] stores the tickets and num[100] stores the grid of the numbers in which the game is played.** | |
| Member functions | | | | |
| Accessibility | | **Name** | | **Details** |
| public | | **void assign();** | | **responsible for assigning values** |
| **void competition();** | | **responsible for detecting winner** |
| **void tickets(int t);** | | **generates tickets** |
| **int play();** | | **it is responsible for starting the game.** |
| **void announcer\_display();** | | **displays the grid of numbers in which the game is to be played.** |
| **void announcer();** | | **assigns the value to num[100]** |
| **int realtime\_play();** | | **it announces a random value.** |
| **void tikets\_of\_players(int t);** | | **displays the tickets** |

**9) snake : public menu\_pointr**

|  |  |  |
| --- | --- | --- |
| Data members | | |
| Accessibility | **Name** | **Details** |
| Private | **char mat[30][60];** | **stores the grid** |
| **int j,i,win,randm,length,ans;** | **i,j-loops**  **win-checks winning**  **randm-choses the maze from the set of three matrix**  **length-stores the length**  **ans-checks whether the previous game is to be played or not.** |
| Member functions | | |
| Accessibility | **Name** | **Details** |
| public | **void outcum();//** | **prints the maze** |
| **int game();** | **this function helps in choosing the options** |
| **void running();** | **the game runs in this function.** |
| **void mat\_pointer();** | **it displays the pointer in the maze in which the game is played.** |
| **void display();** | **the overall output is displayed by this function** |
| **~snake();** | **deletes the linked list** |
| **void snake\_move(int start\_x,int start\_y,int point);** |  |
| **void random\_no\_generate();** | **point position where snake has to go** |
| **int snake\_check(int start\_x,int start\_y);** | **checks whther player out or not** |
| **void ques(int point2);** | **do you want to start prev game** |
| **int ques\_pointer();** | **moves pointer for yes and know** |

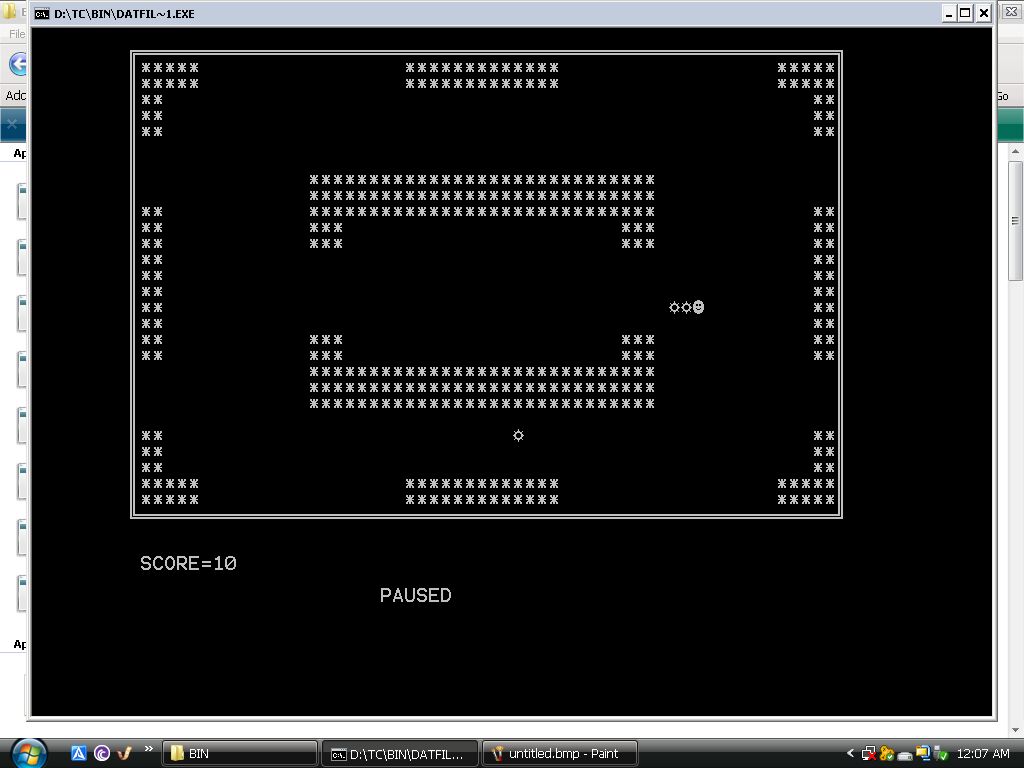
**Structures used:-**

**1) snake\_location**

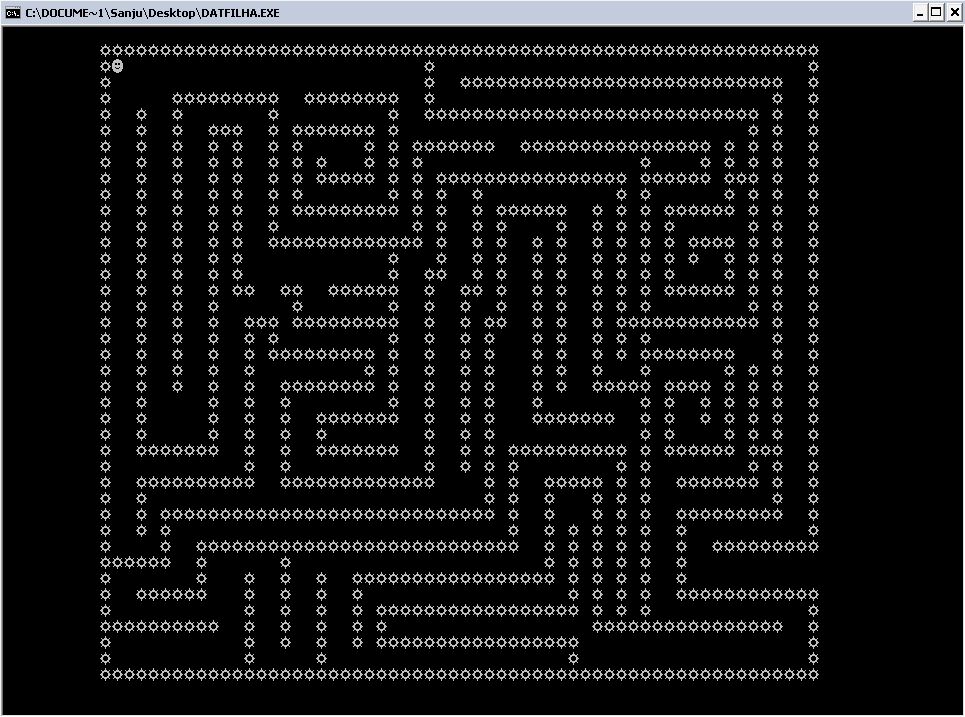
|  |  |  |
| --- | --- | --- |
| Data members | | |
| Accessibility | **Name** | **Details** |
| Public | **int cord\_x;** | **controls the snake size and movement** |
| **snake\_location \*previous;** |
| **int cord\_y;** |
| **snake\_location \*next;** |
| Member functions | | |
| Accessibility | **Name** | **Details** |
| Global objects | **\*front,\*rear;** | **keeps the track of the snake movement and size.** |

**Data files used**

1. **snake.dat**

****

1. **maze.dat**

****

1. **pushboxlvl.dat**

****

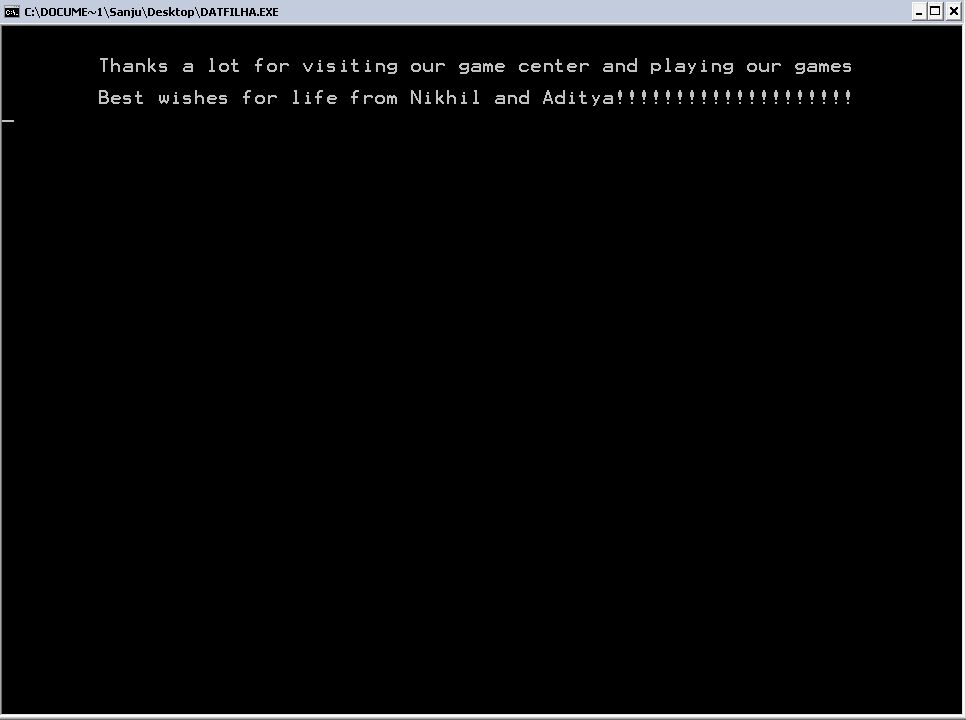
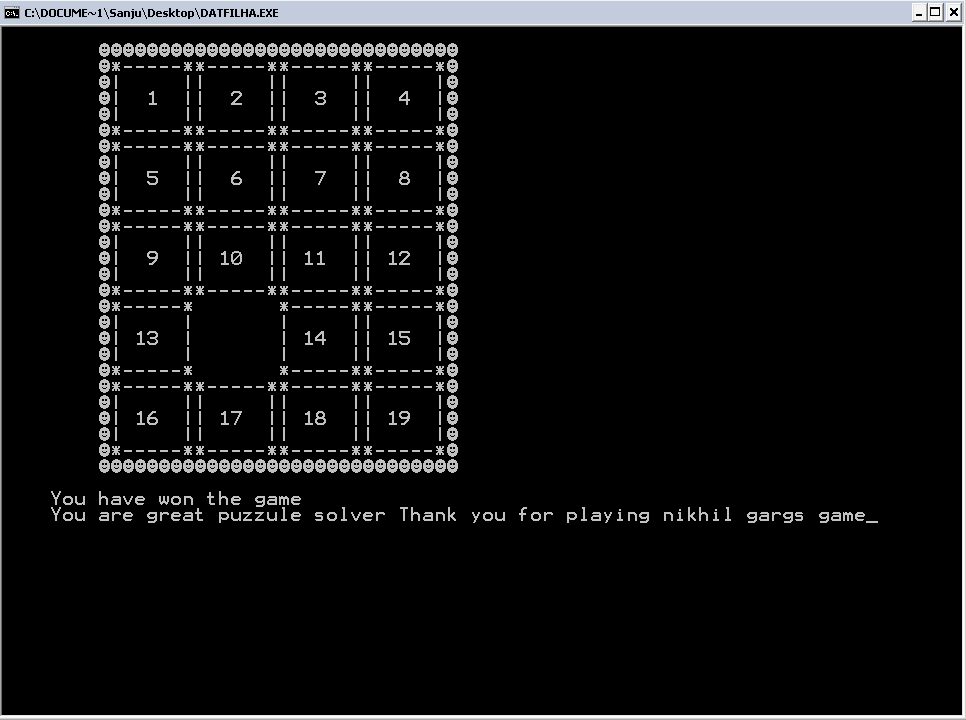
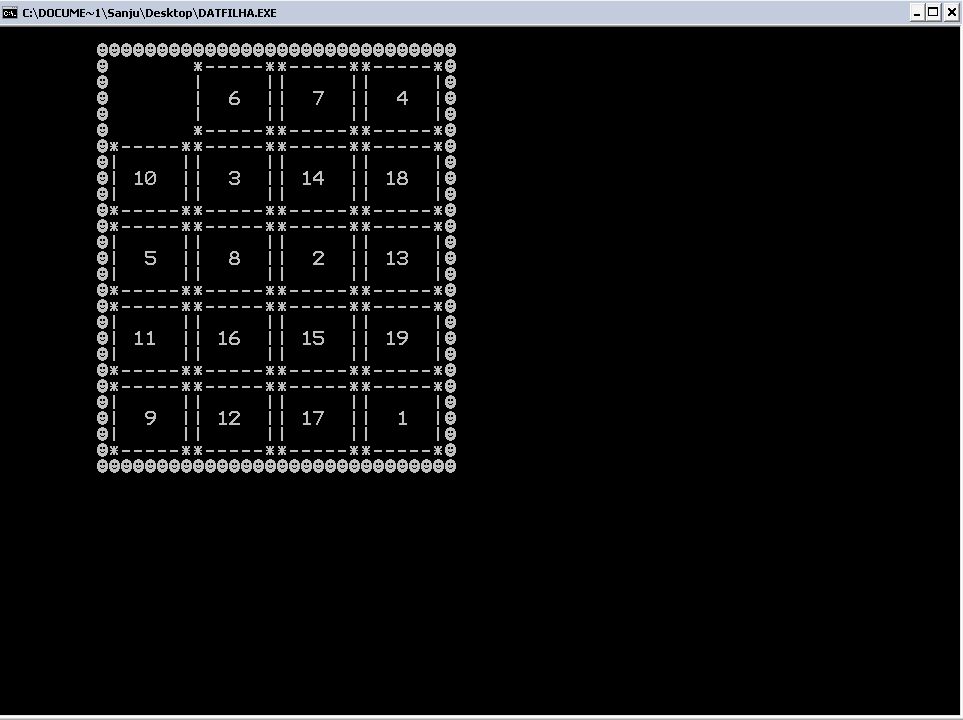
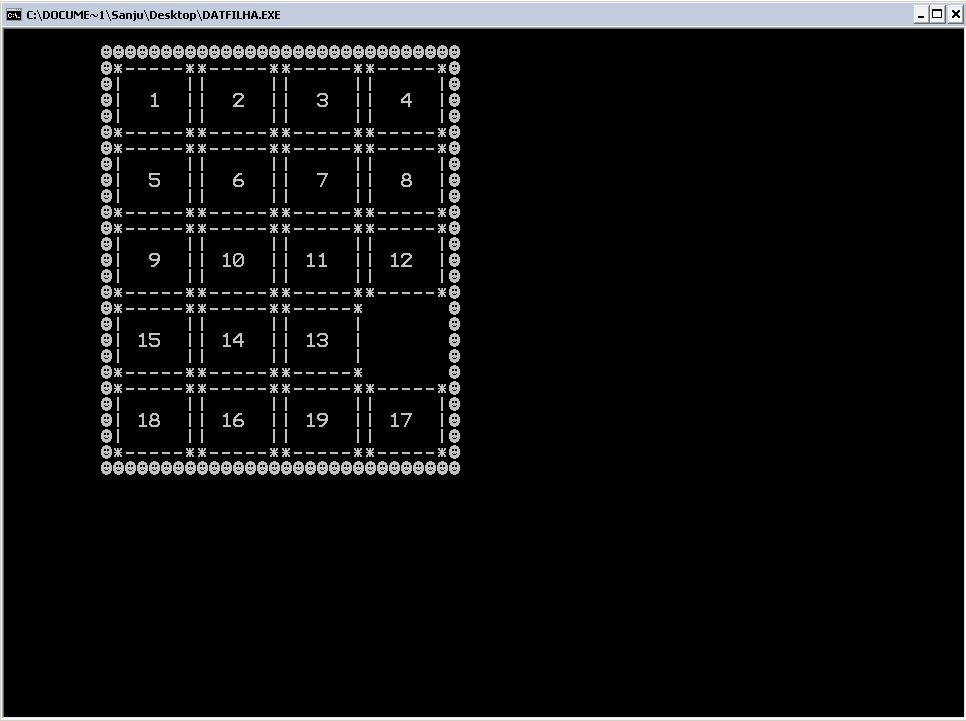
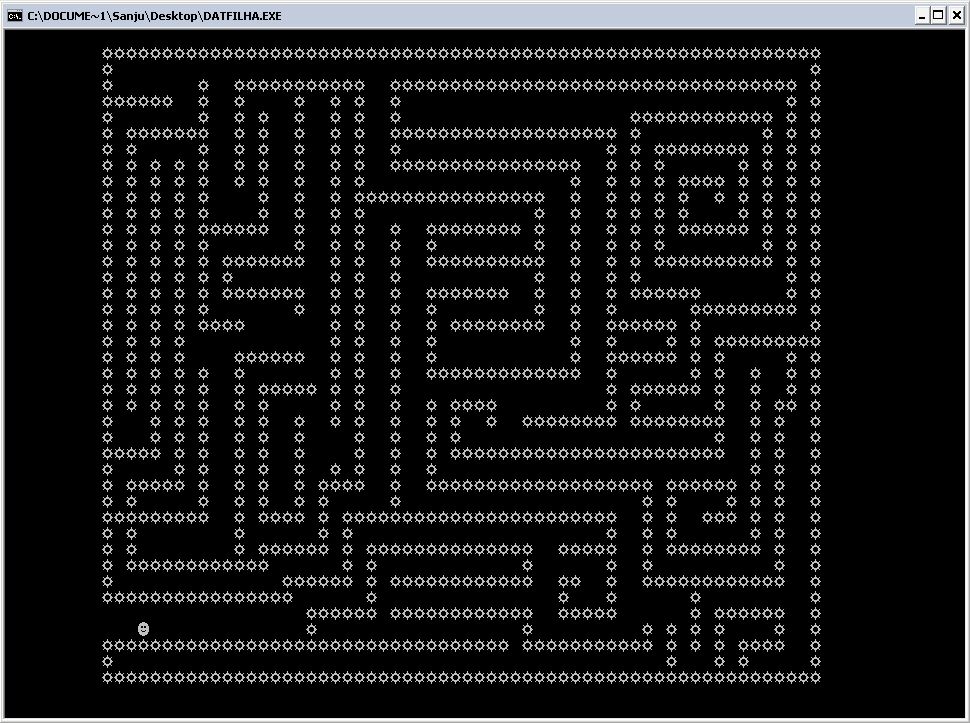
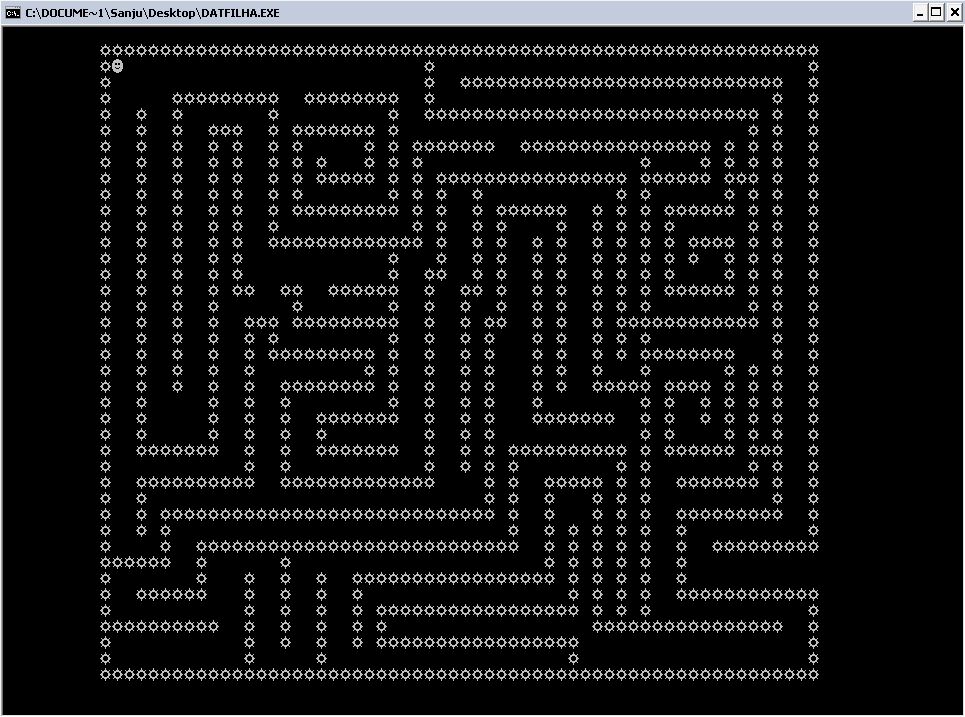
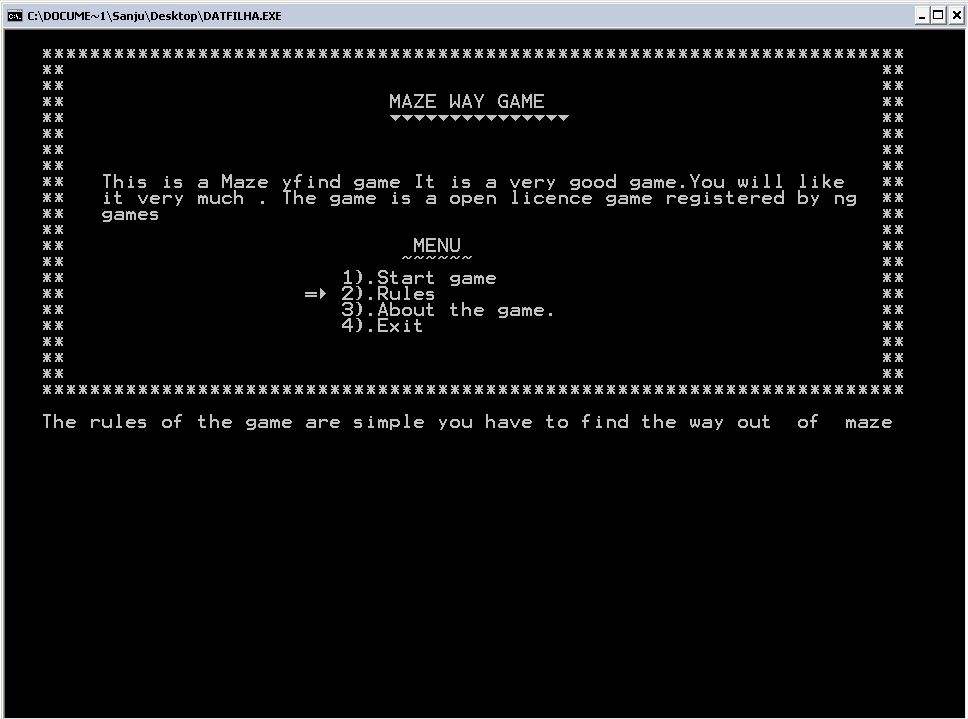
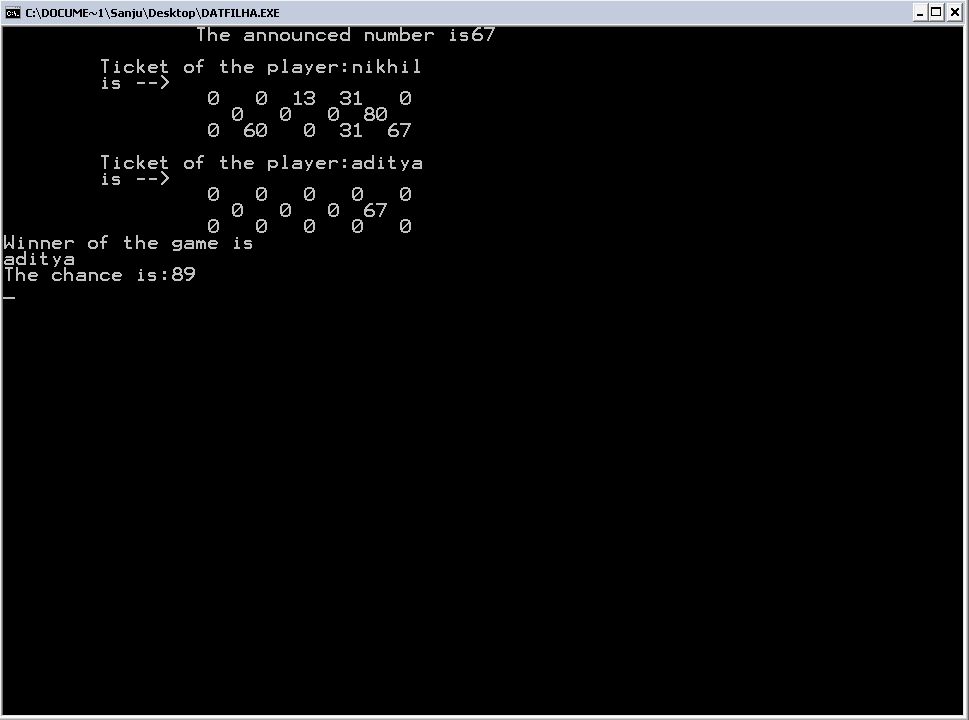
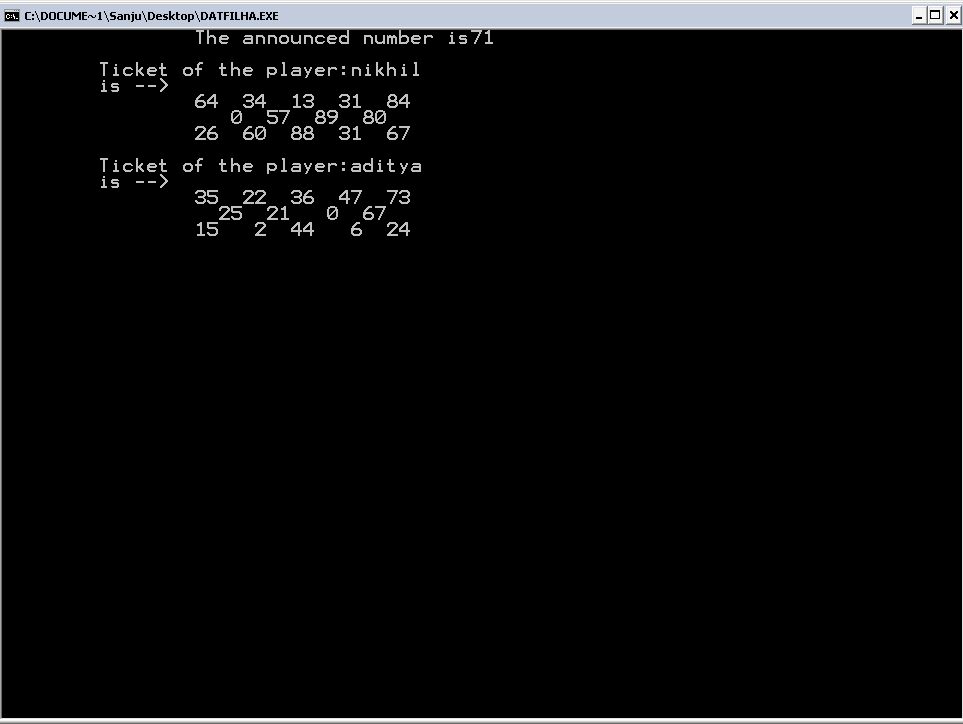
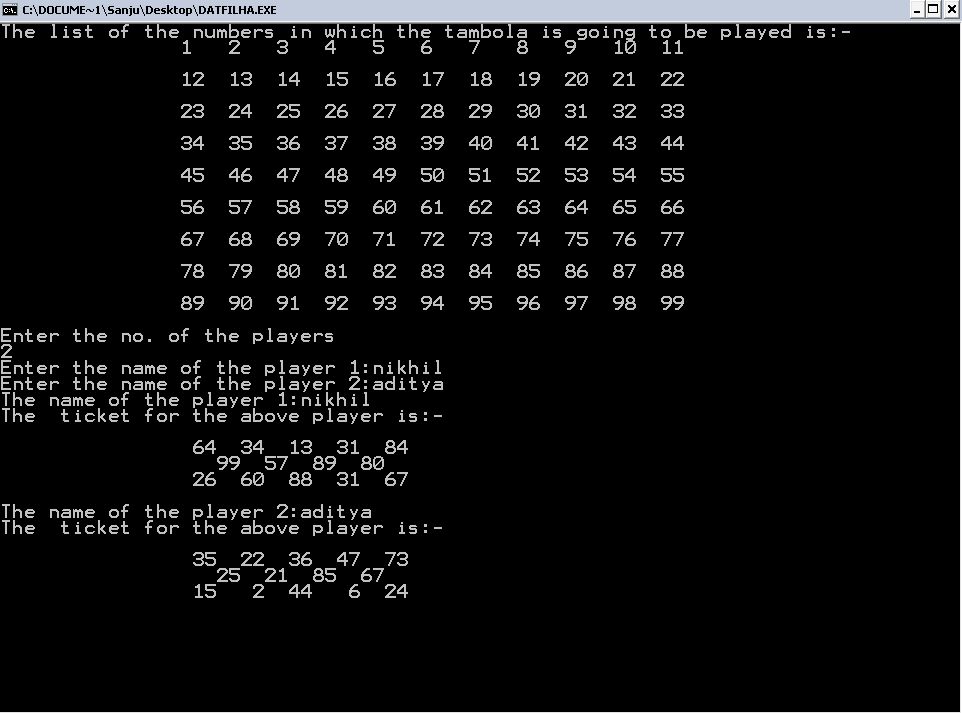
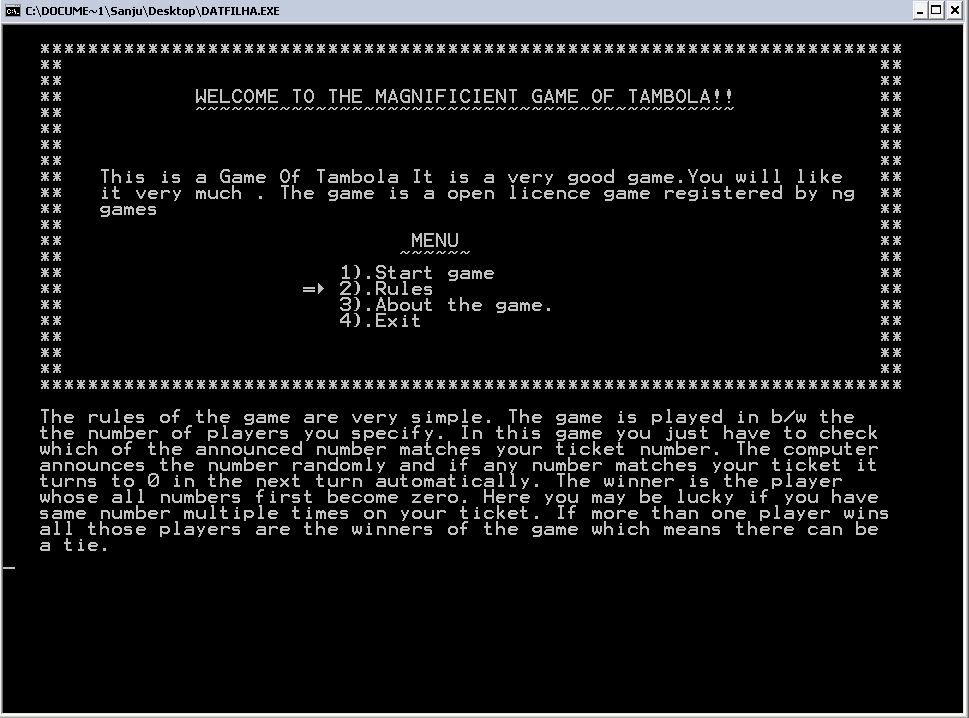
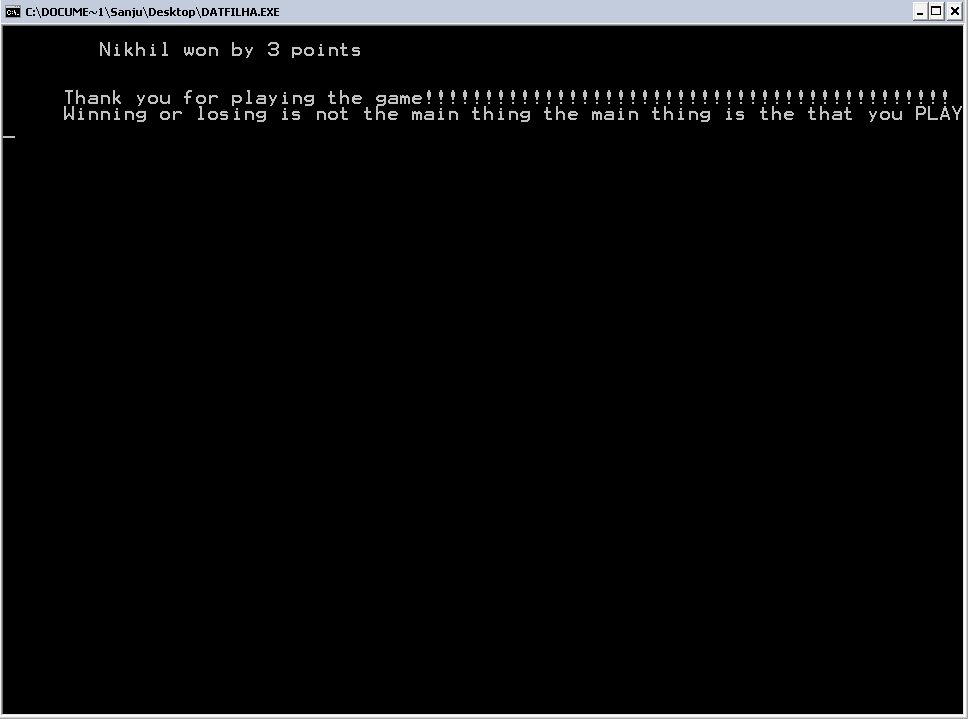
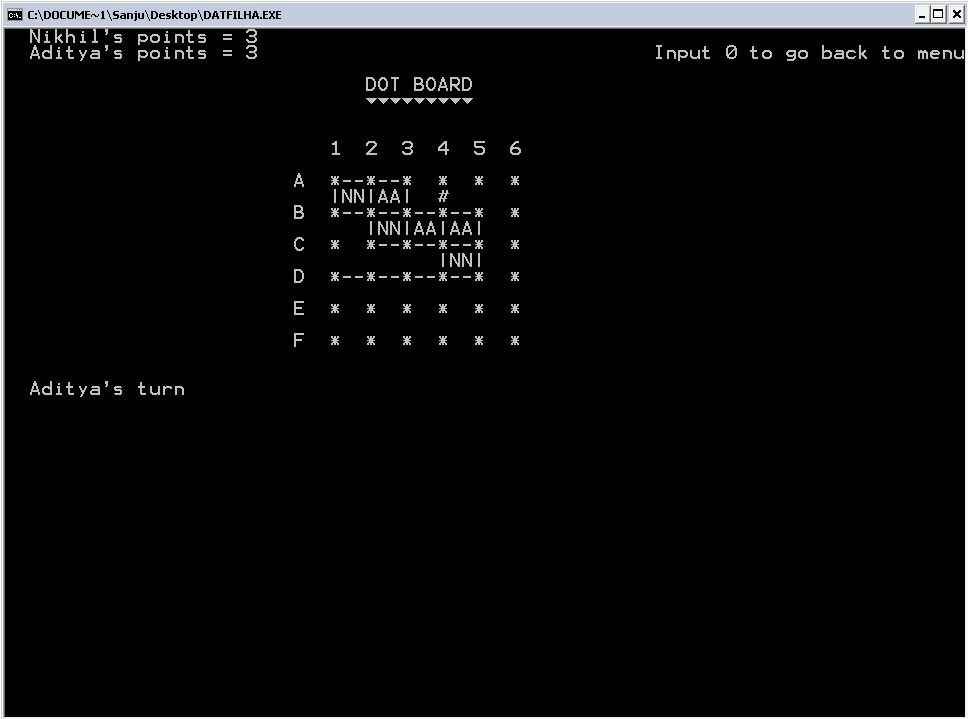
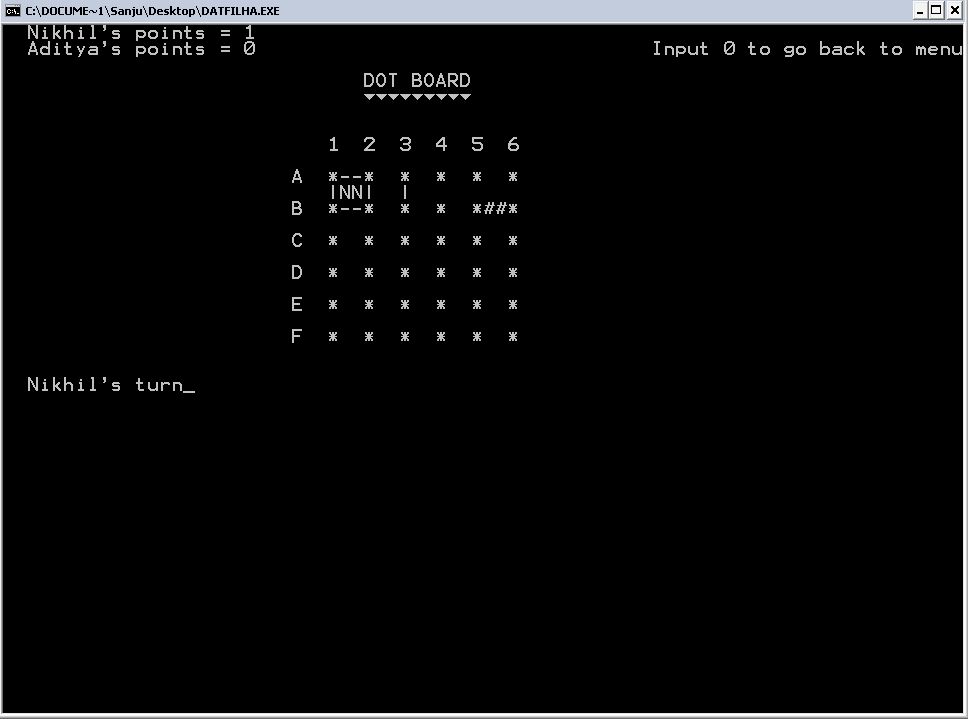
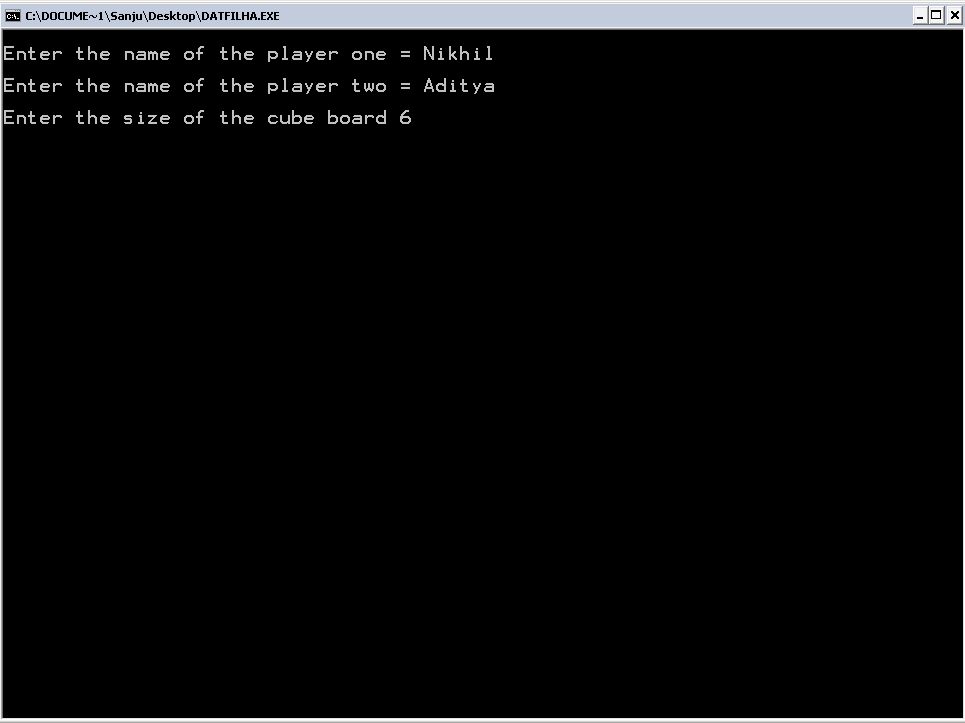
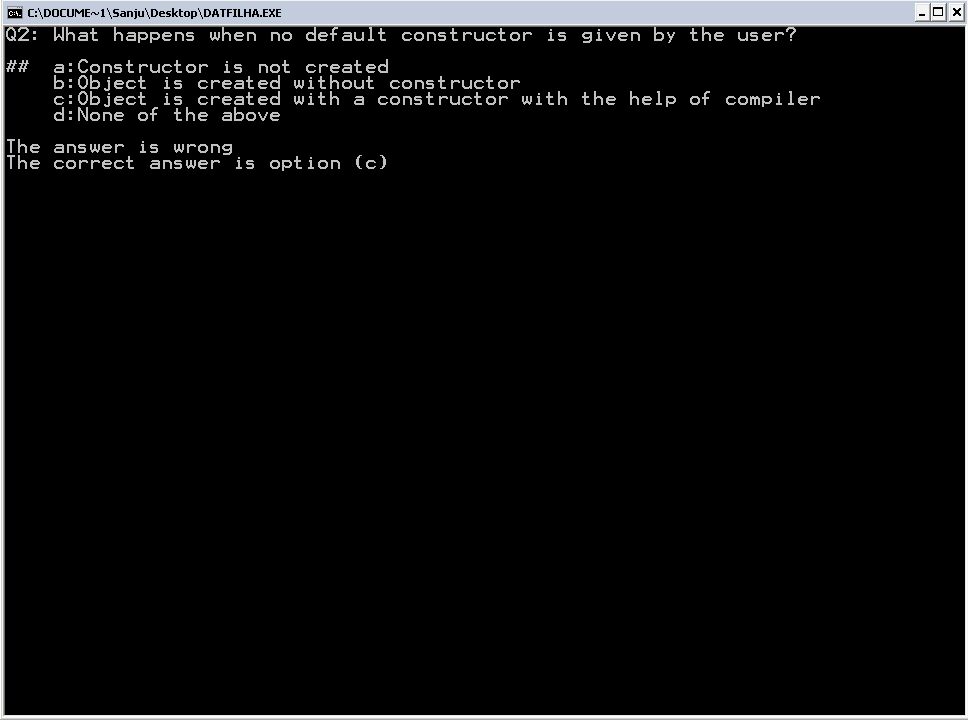
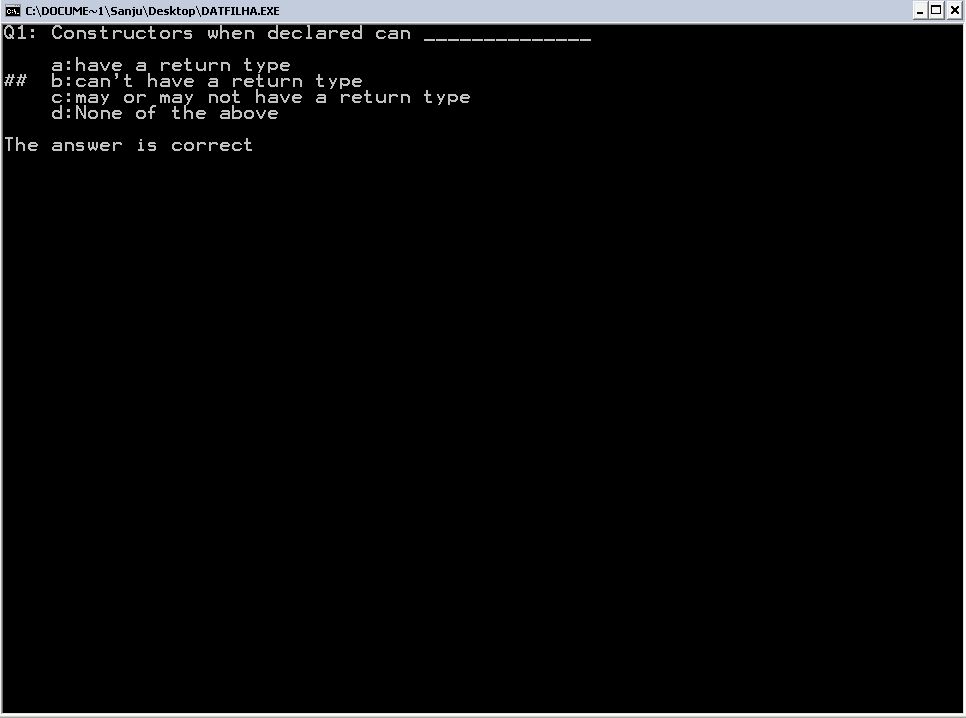
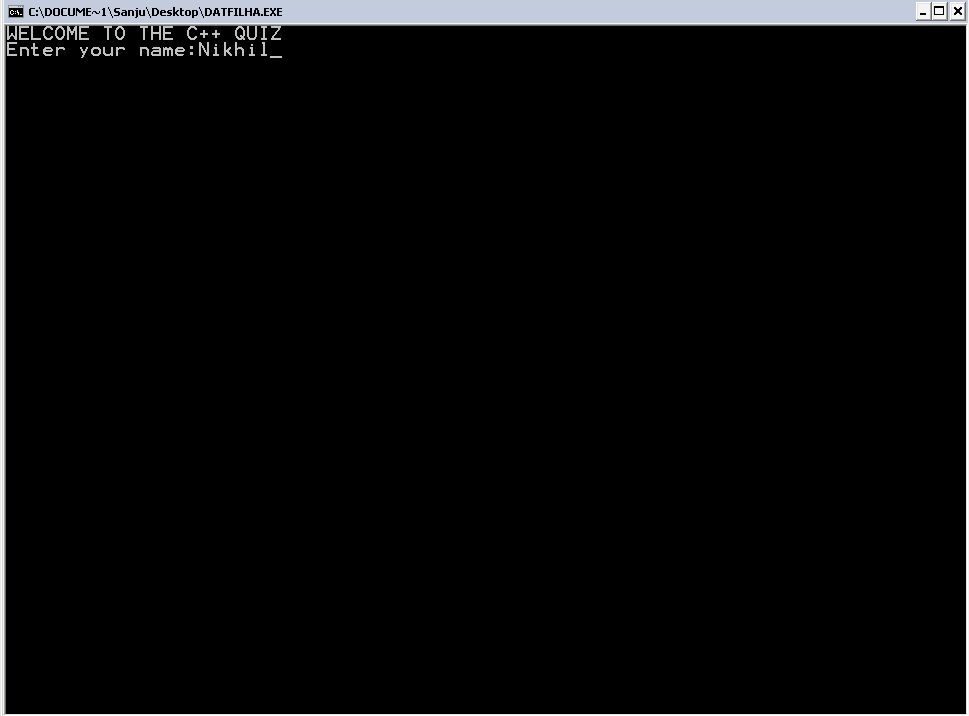
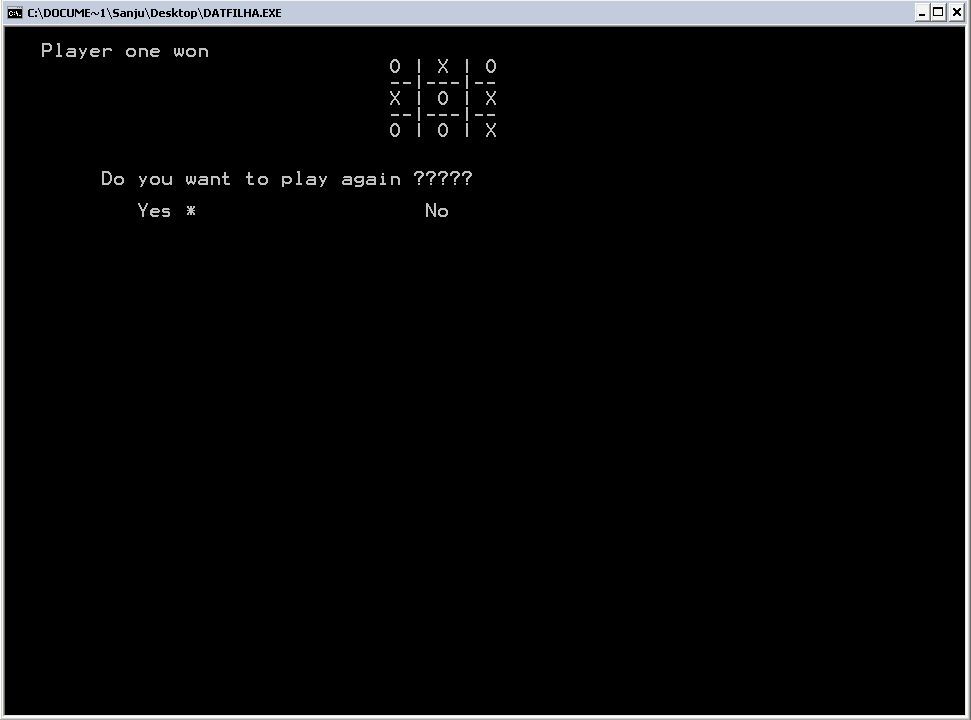


**I would like to express my sincere gratitude to my**

**Computer Science mentor Ms. Neeru Kapoor for her vital support, guidance and encouragement, without which this project would not have come forth.**

**I would also like to express my gratitude to the staff of the Department of Computer Science at Vishal Bharti Public School, New Delhi for their support during the making of this project.**

**Output**



.: Index :.

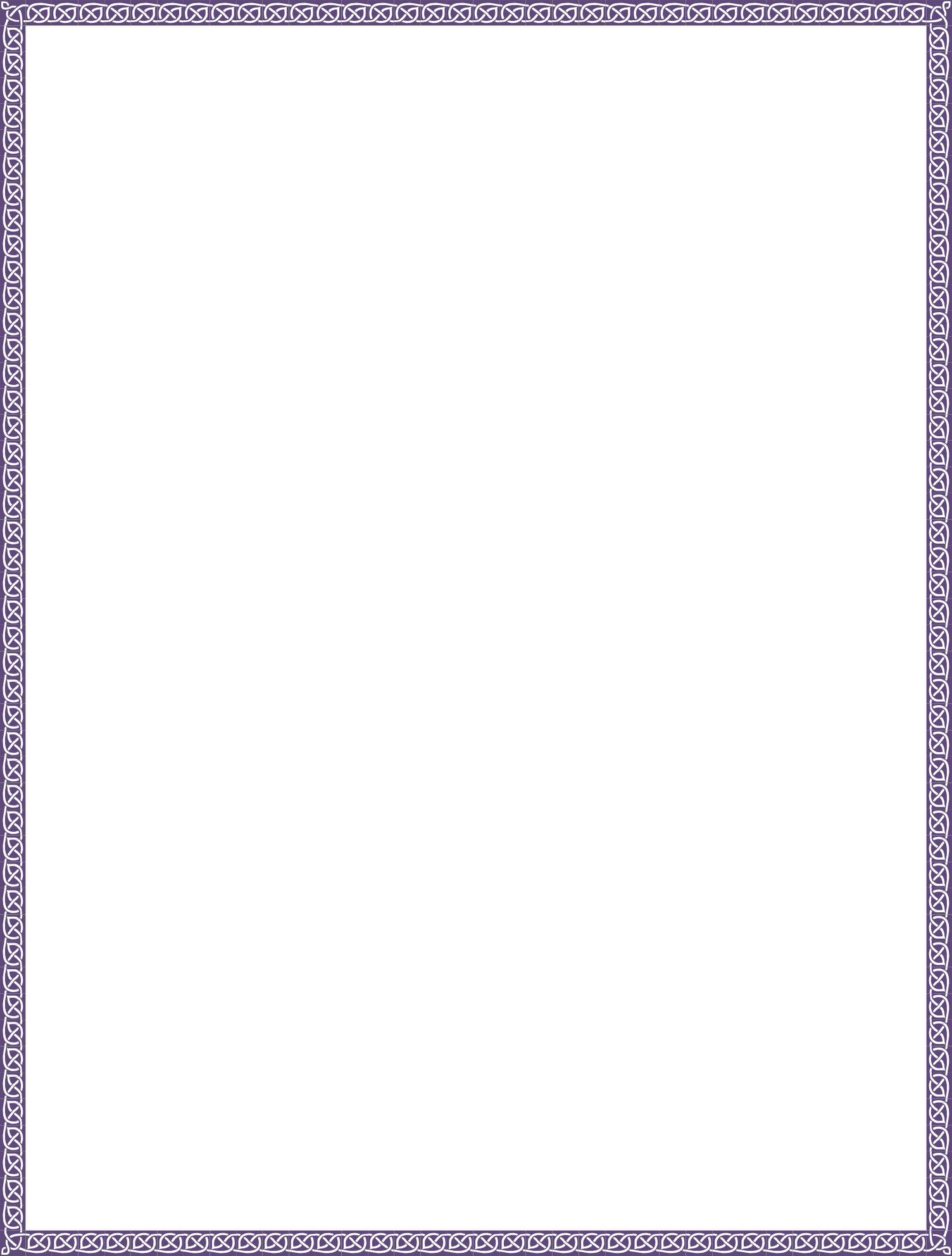
|  |  |
| --- | --- |
| S. No. | TOPIC |
| 1. | Acknowledgement |
| 2. | About the Project  • Introduction  • Data Files Used  • Header Files Used  • Classes Used |
| 3. | Source Code |
| 4. | Output |
| 5. | Enhancements |
| 6. | Bibliography |

Computer Science

Project

By- Aditya choudhary

XII-A



.:Certificate :.

This is to certify that Aditya choudhary, student of **Class XII – A, Vishal Bharti Public School** has completed the project titled **“Game Center”** during the academic year **2015-16** towards partial fulfillment of credit for

the Computer Science Practical evaluation of

**CBSE 2015-16** under my supervision.

**Ms. Neeru Kapoor**

**(HOD – Computer Science)** Department of Computer Science Vishal Bharti Public School, New Delhi