**GLOBAL ACADEMY OF TECHNOLOGY**

**Department of Computer Science and Engineering**

**(Accredited by NBA 2019-2022)**

**Rajarajeshwari Nagar, Bengaluru – 560 098**

**TUTORIAL CLASS DATA STRUCTURES**

**Project Name: SUDOKU GAME**

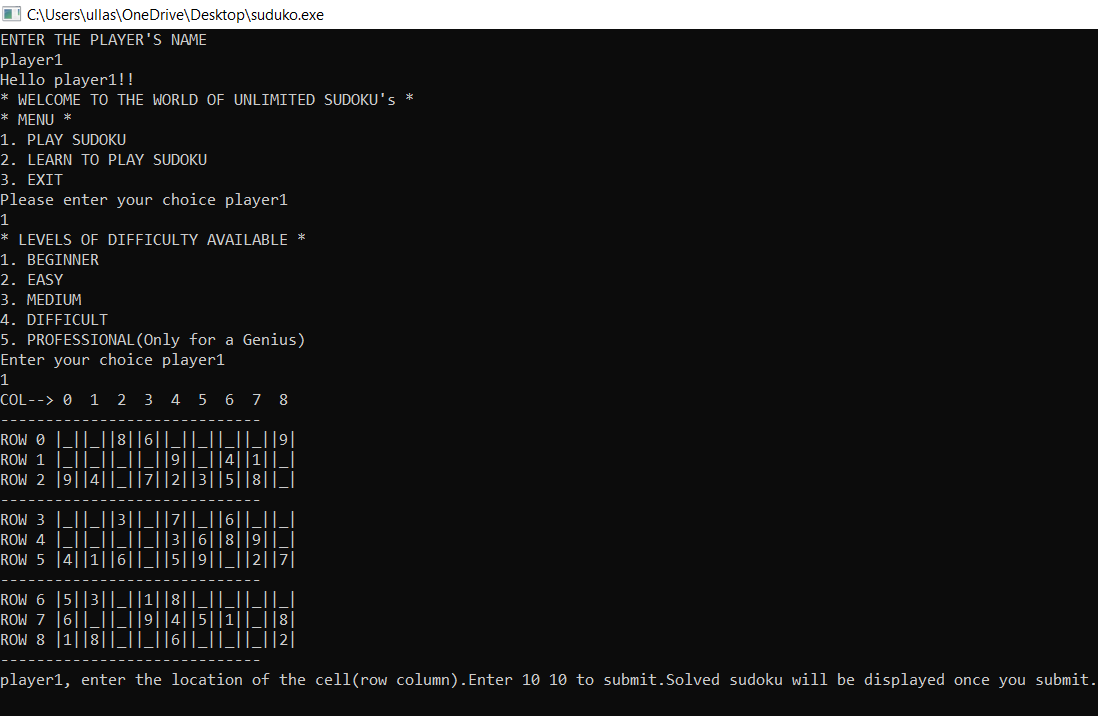
**Name & USN of Team members:**

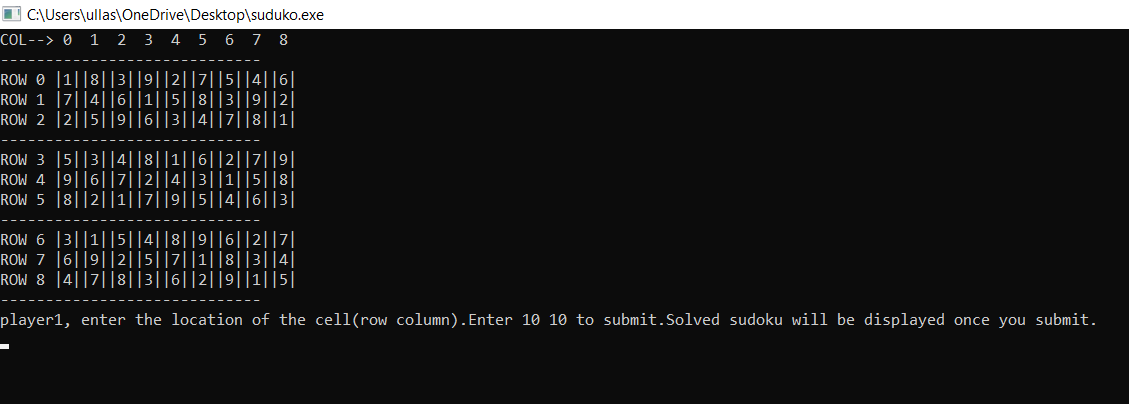
1. **VIKAS KR (1GA20CS168)**
2. **SRAJAN MP (1GA20CS141)**
3. **SAGAR K GOWDA (1GA20CS121)**
4. **CHANDANA BC (1GA2EE008)**
5. **USHA N (1GA20CS158)**

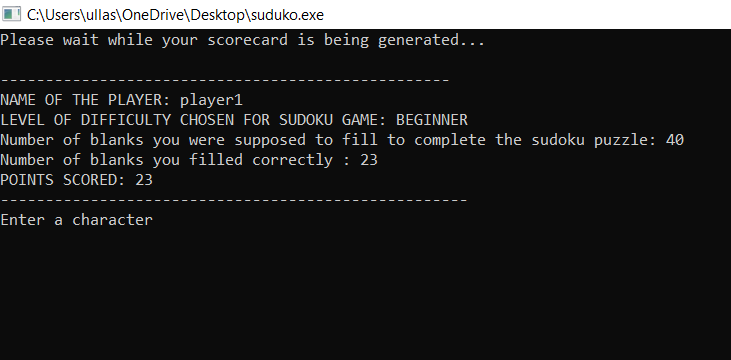
**Project Description :**

Sudoku is a game made up of 81 cells organized into 9 rows, 9 columns, and 9 boxes. A puzzle will start with 17 or more cells already filled in. The challenge is to fill in the rest. To solve a puzzle, the user places I through 9 in each row, each column, and each minigrid such that no number repeats itself in that particular row, column or minigrid..

**Snap-shots of Outputs**







**Code:**

**#include<stdio.h>**

**#include<stdlib.h>**

**#include<string.h>**

**int i,j,p,q,z,k=0,r,e=0,counter,del,gen,points,hintmat[9][9],copy1,copy2,ques[9][9],check,ctr1,ctr2,hint,cl,ctr=0,l=0,c,o,d,b,value,a[9][9],m[50],v[50],w[50],n[50],org[9][9],finalctr=0,choice,colour,level,helpexit=0,blank,store,hintcount,t,u;**

**char name[50];**

**float time1;**

**void generator(void);**

**void scorecard(void);**

**void play(void);**

**void print(void);**

**void help(void);**

**void sudoku(void);**

**void game(void);**

**void print()**

**{**

**printf("COL--> 0 1 2 3 4 5 6 7 8");**

**printf ("\n-----------------------------\n");**

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

**{**

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

**{ if(q==0)**

**{**

**printf("ROW %d ",p);**

**if(a[p][q]==0)**

**{**

**printf("|");**

**printf("\_");**

**printf("|");**

**}**

**else**

**{printf("|");**

**printf("%d",a[p][q]);**

**printf("|");}**

**}**

**else if (q==2||q==5)**

**{**

**if(a[p][q]==0)**

**{**

**printf("|\_|");}**

**else**

**{**

**printf ("|%d",a[p][q]);**

**printf("|");**

**}**

**}**

**else**

**{**

**if(a[p][q]==0)**

**{**

**printf("|\_|");**

**}**

**else**

**{**

**printf("|%d",a[p][q]);**

**printf("|");}**

**}**

**}**

**if (p==2||p==5||p==8)**

**{**

**printf ("\n-----------------------------");**

**}**

**printf("\n");**

**}**

**}**

**void sudoku()**

**{**

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

**{**

**if(i==0)**

**{**

**break;**

**}**

**if(a[k][j]==a[i][j])**

**{**

**a [i][j]=(rand()%9)+1;**

**ctr++;**

**if(ctr==100)**

**{**

**ctr=0;**

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

**a[i][j]=0;**

**j=0;**

**a[i][j]=(rand()%9)+1;**

**sudoku();**

**}**

**sudoku();**

**}**

**}**

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

**{**

**if(j==0)**

**{**

**break;**

**}**

**if(a[i][j]==a[i][l])**

**{**

**a [i][j]=(rand()%9)+1;**

**sudoku();//RECURSION**

**}**

**}**

**o=0;**

**for (c=0;c<=6;c+=3)**

**{if(i>=c&&i<(c+3))**

**{**

**r=c;**

**break;**

**}**

**}**

**for(d=0;d<=6;d+=3)**

**{**

**if (j>=d&&j<(d+3))**

**{**

**cl=d;**

**break;**

**}**

**}**

**for (r=c;r<(c+3);r++)**

**{**

**for(cl=d;cl<(d+3);cl++)**

**{**

**if(cl==j&&r==i)**

**{o=1;**

**break;**

**}**

**if(a[r][cl]==a[i][j])**

**{**

**a[i][j]=(rand()%9)+1;**

**sudoku();**

**}**

**}**

**if(o==1)**

**break;**

**}**

**}**

**void dighole()**

**{**

**for(z=0;z<blank;z++)**

**{**

**e=0;**

**i=rand()%9;**

**m[z]=i;**

**j=rand()%9;;**

**n[z]=j;**

**if(z!=0)**

**{**

**for(l=2;l<z;l++)**

**{**

**if(i==m[l]&&j==n[l])**

**{**

**z=z-1;**

**e=1;**

**break;**

**}**

**}**

**}**

**if(e!=1)**

**a[i][j]=0;**

**}**

**for(copy1=0;copy1<9;copy1++)**

**{**

**for(copy2=0;copy2<9;copy2++)**

**{**

**ques[copy1][copy2]=a[copy1][copy2];**

**}**

**}**

**}**

**void help()**

**{ system("cls");**

**printf("\t\*RULES AND INSTRUCTIONS\*\*\n");**

**printf("Hi %s, welcome to sudoku help. Even if you are completely new to sudoku, you can learn it from the beginning here.\n",name);**

**printf("Sudoku is a game of numbers which will test your intelligence. Basically,it consists of a 9x9 grid.");**

**printf("The grid will have certain numbers which are already given and serve as clues to the player.You have to fill the grid");**

**printf("with every row and column consisting of numbers from 1 to 9.Every 3x3 block must also consist of numbers from 1 to 9.\n");**

**printf("So, %s, you will have to complete the entire 9x9 grid of 81 numbers without violating these rules.",name);**

**printf("And %s,you will have to look into the locations and give the location of the cell you want to enter the number into.",name);**

**printf("For example, 0 2 or 3 4 etc.\n");**

**printf("There will be 5 levels of difficulty.In the beginner mode,for each correct answer 1 point is given.In the easy mode,2 points,in the medium mode 3 points");**

**printf(",in the difficult mode 4 points and in the professional mode 5 points are given.\n");**

**printf("That's all are the rules for playing this simple game. Enjoy playing sudoku, i will guarantee you that you will have a great time playing sudoku here.\n");**

**printf("Enter a character to see a sample sudoku\n");**

**getch();**

**system("cls");**

**printf("HERE IS A SAMPLE SUDOKU FILLED GRID\n");**

**print();**

**printf("Enter a character to continue once you have seen the grid and return to previous menu.\n");**

**getch();**

**system("cls");**

**helpexit=1;**

**}**

**void game()**

**{**

**system("cls");**

**if(helpexit!=1)**

**{**

**printf("ENTER THE PLAYER'S NAME\n");**

**gets(name);**

**printf("Hello %s!!\n",name);**

**}**

**do{**

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

**{**

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

**{**

**hintmat[t][u]=0;**

**}**

**}**

**hintcount=0;**

**generator();**

**printf("\* WELCOME TO THE WORLD OF UNLIMITED SUDOKU's \*\n");**

**printf("\* MENU \*\n");**

**printf("1. PLAY SUDOKU\n");**

**printf("2. LEARN TO PLAY SUDOKU\n");**

**printf("3. EXIT\n");**

**printf("Please enter your choice %s \n",name);**

**scanf("%d",&choice);**

**switch(choice)**

**{**

**case 1: printf("\* LEVELS OF DIFFICULTY AVAILABLE \*\n");**

**printf("1. BEGINNER\n");**

**printf("2. EASY\n");**

**printf("3. MEDIUM\n");**

**printf("4. DIFFICULT\n");**

**printf("5. PROFESSIONAL(Only for a Genius)\n");**

**printf("Enter your choice %s \n",name);**

**scanf("%d",&level);**

**switch(level)**

**{**

**case 1: blank=40;**

**dighole();**

**play();**

**scorecard();**

**break;**

**case 2: blank=50;**

**dighole();**

**play();**

**scorecard();**

**break;**

**case 3: blank=56;**

**dighole();**

**play();**

**scorecard();**

**break;**

**case 4: blank=60;**

**dighole();**

**play();**

**scorecard();**

**break;**

**case 5: blank=65;**

**dighole();**

**play();**

**scorecard();**

**break;**

**default:printf("Please choose the proper level of difficulty %s.You will be redirected to previous menu \n",name);**

**break;**

**}**

**break;**

**case 2: help();**

**break;**

**case 3: printf("THANK YOU FOR PLAYING %s !! Come again....\n",name);**

**getch();**

**exit(0);**

**default:printf("Please enter the proper choice number %s\n",name);**

**break;**

**}**

**}while(choice!=3);**

**}**

**void generator()**

**{**

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

**{**

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

**{**

**a[i][j]=0;**

**}**

**}**

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

**{**

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

**{**

**b=rand();**

**value=(b%9)+1;**

**a[i][j]=value;**

**if ((i==j)&&(i==0))**

**{if ((k==l)&&(k==0))**

**{**

**continue;**

**}**

**}**

**else**

**{**

**sudoku();**

**}**

**}**

**}**

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

**{**

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

**{**

**org[i][j]=a[i][j];**

**}**

**}**

**}**

**void play()**

**{**

**print();**

**while(1)**

**{**

**gen=0;**

**hint=-1;**

**printf("%s, enter the location of the cell(row column).Enter 10 10 to submit.Solved sudoku will be displayed once you submit.\n",name);**

**scanf("%d%d",&i,&j);**

**if(i<=8&&j<=8)**

**{**

**if(ques[i][j]!=0)**

**{gen=1;**

**printf("This is a generated element %s. You cannot change it.Enter a character to continue.\n",name);**

**getch();**

**system("cls");**

**print();**

**}**

**if(gen!=1)**

**{**

**if(hintmat[i][j]!=0)**

**{ printf("You have taken a hint for this element.Do not try to change it %s.Enter a character to continue.\n",name);**

**getch();**

**system("cls");**

**print();**

**continue;**

**}**

**v[counter]=i;**

**w[counter]=j;**

**counter++;**

**printf("If you want a hint,type 0 otherwise type 1\n");**

**scanf("%d",&hint);**

**if(hint==0)**

**{**

**hintcount++;**

**a[i][j]=org[i][j];**

**hintmat[i][j]=a[i][j];**

**system("cls");**

**print();**

**}**

**else if(hint==1)**

**{**

**printf("Enter a number from 1-9 or enter 0 to erase.\n");**

**scanf("%d",&store);**

**if(store==0&&hintmat[i][j]==0)**

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

**system("cls");**

**print();**

**continue;**

**}**

**if(store>=1&&store<=9)**

**{**

**a[i][j]=store;**

**if(store==org[i][j])**

**{**

**points++;**

**}**

**system("cls");**

**print();**

**}**

**else**

**{ printf("\*\*Enter a valid number %s\*\*\n",name);**

**printf("Enter any character to continue.If you have any doubts,refer help.\n");**

**getch();**

**system("cls");**

**print();**

**}**

**}**

**}**

**}**

**else**

**{if(i==10&&j==10)**

**break;**

**else**

**{**

**printf("INVALID ELEMENT LOCATION ENTERED!\n");**

**system("cls");**

**print();**

**}**

**}**

**}**

**printf("Thank you for playing %s ",name);**

**}**

**void scorecard()**

**{ system("cls");**

**if(points==blank)**

**printf("THAT'S GREAT!!..You solved the entire sudoku correctly..You played really well %s\n",name);**

**if(level==2)**

**points\*=2;**

**else if(level==3)**

**points\*=3;**

**else if(level==4)**

**points\*=4;**

**else if(level==5)**

**points\*=5;**

**printf("Please wait while your scorecard is being generated...\n");**

**for(time1=0;time1<5000000;time1++)**

**;**

**printf("\n--------------------------------------------------\n");**

**printf("NAME OF THE PLAYER: %s \n",name);**

**if(level==1)**

**{ printf("LEVEL OF DIFFICULTY CHOSEN FOR SUDOKU GAME: BEGINNER \n");**

**printf("Number of blanks you were supposed to fill to complete the sudoku puzzle: 40\n");**

**printf("Number of blanks you filled correctly : %d\n",points);**

**if(points>30)**

**printf("YOU PLAYED REALLY WELL %s\n",name);**

**}**

**else if(level==2)**

**{ printf("LEVEL OF DIFFICULTY CHOSEN FOR SUDOKU GAME: EASY \n");**

**printf("Number of blanks you were supposed to fill to complete the sudoku puzzle: 50\n");**

**printf("Number of blanks you filled correctly : %d\n",(points)/2);**

**if(points>80)**

**printf("YOU PLAYED REALLY WELL %s\n",name);**

**}**

**else if(level==3)**

**{ printf("LEVEL OF DIFFICULTY CHOSEN FOR SUDOKU GAME: MEDIUM \n");**

**printf("Number of blanks you were supposed to fill to complete the sudoku puzzle: 56\n");**

**printf("Number of blanks you filled correctly : %d\n",(points)/3);**

**if(points>85)**

**printf("YOU PLAYED REALLY WELL %s\n",name);**

**}**

**else if(level==4)**

**{ printf("LEVEL OF DIFFICULTY CHOSEN FOR SUDOKU GAME: DIFFICULT \n");**

**printf("Number of blanks you were supposed to fill to complete the sudoku puzzle: 60\n");**

**printf("Number of blanks you filled correctly : %d\n",(points)/4);**

**if(points>100)**

**printf("YOU PLAYED REALLY WELL %s\n",name);**

**}**

**else if(level==5)**

**{ printf("LEVEL OF DIFFICULTY CHOSEN FOR SUDOKU GAME: PROFESSIONAL \n");**

**printf("Number of blanks you were supposed to fill to complete the sudoku puzzle: 65\n");**

**printf("Number of blanks you filled correctly : %d\n",(points)/5);**

**if(points>110)**

**printf("YOU PLAYED REALLY WELL %s\n",name);**

**}**

**printf("POINTS SCORED: %d\n",points);**

**if(points>0)**

**printf("----------------------------------------------------\n");**

**printf("Enter a character\n");**

**getch();**

**system("cls");**

**printf("YOUR SUDOKU\n");**

**print();**

**for(time1=0;time1<10000000;time1++)**

**;**

**printf("SOLVED SUDOKU\n");**

**for(ctr1=0;ctr1<9;ctr1++)**

**{**

**for(ctr2=0;ctr2<9;ctr2++)**

**{**

**a[ctr1][ctr2]=org[ctr1][ctr2];**

**}**

**}**

**}**

**int main()**

**{**

**srand(time(NULL));/\*The random function rand() which if not**

**seeded will return the same random numbers starting with 41\*/**

**generator();**

**game();**

**return 0;**

**}**