CSE 2003: Lab Assignment #11

Due on Thursday, April 13, 2017

Prof. Shaik Naseera 2:00pm

Jacob John

Contents

Problem 1

Write a program to generate solution for N queens problem using Backtracking approach using C

Listing 1: N queens problem using Backtracking approach using C

```c
#include<stdio.h>
#include<stdlib.h>

int board[20],count;

int main()
{
 int n,i,j;
 void queen(int row,int n);

 printf(" - N Queens Problem Using Backtracking -");
 printf("\n\nEnter number of Queens:");
 scanf("%d",&n);
 queen(1,n);
 return 0;
}

//function for printing the solution
void print(int n)
{
 int i,j;
 printf("\n\nSolution %d:\n\n",++count);

 for(i=1;i<=n;++i)
  printf("\t%d",i);

 for(i=1;i<=n;++i)
 {
  printf("\n\n%d",i);
  for(j=1;j<=n;++j) //for nxn board
  {
   if(board[i]==j)
    printf("\tQ"); //queen at i,j position
   else
    printf("\t-"); //empty slot
  }
 }
}

/*funtion to check conflicts
If no conflict for desired postion returns 1 otherwise returns 0*/
int place(int row,int column)
{
 int i;
 for(i=1;i<=row-1;++i)
 {
  //checking column and digonal conflicts
```

Problem 1 continued on next page...

```
  if(board[i]==column)
   return 0;
  else
   if(abs(board[i]-column)==abs(i-row))
    return 0;
 }

 return 1; //no conflicts
}

//function to check for proper positioning of queen
void queen(int row,int n)
{
 int column;
 for(column=1;column<=n;++column)
 {
  if(place(row,column))
  {
   board[row]=column; //no conflicts so place queen
   if(row==n) //dead end
    print(n); //printing the board configuration
   else //try queen with next position
    queen(row+1,n);
  }
 }
}
```

Output:

```
 - N Queens Problem Using Backtracking -

Enter number of Queens:4

Solution 1:

       1      2      3      4

1      -      Q      -      -

2      -      -      -      Q

3      Q      -      -      -

4      -      -      Q      -

Solution 2:

       1      2      3      4

1      -      -      Q      -

2      Q      -      -      -

3      -      -      -      Q

4      -      Q      -      -Jacobs-MacBook-Pro:Downloads jacobjohn$ ./a.o
ut
 - N Queens Problem Using Backtracking -

Enter number of Queens:1

Solution 1:

       1

1      QJacobs-MacBook-Pro:Downloads jacobjohn$ clear

Jacobs-MacBook-Pro:Downloads jacobjohn$ ./a.out
 - N Queens Problem Using Backtracking -

Enter number of Queens:4
```

n_queen_backtracking.c | knapsack_greedy.c

```
#include<stdio.h>
#include<stdlib.h>

int board[20],count;

int main()
{
 int n,i,j;
 void queen(int row,int n);

 printf(" - N Queens Problem Using Backtracking -");
 printf("\n\nEnter number of Queens:");
 scanf("%d",&n);
 queen(1,n);
 return 0;
}

//function for printing the solution
void print(int n)
{
 int i,j;
 printf("\n\nSolution %d:\n\n",++count);

 for(i=1;i<=n;++i)
  printf("\t%d",i);

 for(i=1;i<=n;++i)
 {
  printf("\n\n%d",i);
  for(j=1;j<=n;++j) //for nxn board
  {
   if(board[i]==j)
    printf("\tQ"); //queen at i,j position
   else
    printf("\t-"); //empty slot
  }
 }
}

/*funtion to check conflicts
If no conflict for desired postion returns 1 otherwise returns 0*/
int place(int row,int column)
{
 int i;
 for(i=1;i<=row-1;++i)
 {
```

Line: 16:2 | C | Tab Size: 4 | main