School of Computer Science and Engineering (SCOPE)

Assessment – 3

Design and analysis of Algorithm (Lab Component)

MCSE502L

Course Name : Design and analysis of Algorithm (Lab Component)
Course Code : MCSE502L
Slot: L35+L36
Course instructor: MOHAMMAD ARIF

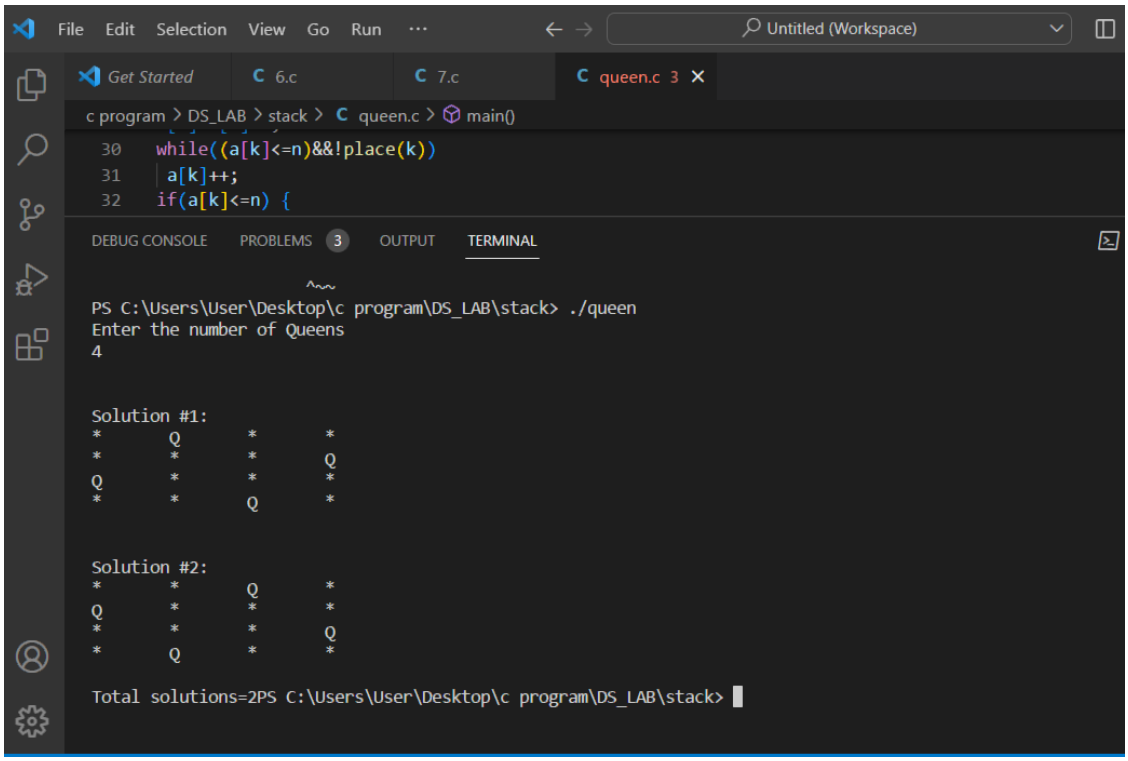
Name : NIDHI SINGH
Registration no. : 22MAI0015

1. Implement 4-Queen Problem.

Code :-

```
#include<stdio.h>
#include<math.h>
int a[30],count=0;
int place(int pos) {
int i;
for (i=1;i<pos;i++) {
if((a[i]==a[pos])||((abs(a[i]-a[pos])==abs(i-pos))))
 return 0;
}
return 1;
}
void print_sol(int n) {
int i,j;
count++;
printf("\n\nSolution #%d:\n",count);
for (i=1;i<=n;i++) {
for (j=1;j<=n;j++) {
if(a[i]==j)
 printf("Q\t"); else
 printf("*\t");
}
printf("\n");
}
}
void queen(int n) {
int k=1;
a[k]=0;
while(k!=0) {
a[k]=a[k]+1;
while((a[k]<=n)&&!place(k))
 a[k]++;
if(a[k]<=n) {
if(k==n)
 print_sol(n); else {
k++;
a[k]=0;
}
} else
 k--;
}
}
void main() {
int i,n;
printf("Enter the number of Queens\n");
scanf("%d",&n);
queen(n);
printf("\nTotal solutions=%d",count);
}
```

Output :-



2. Implement String Matching Algorithms: Rabin Karp Algorithm, KMP Algorithm

Rabin Karp Algorithm:-

```c
#include<stdio.h>
#include<string.h>
// d is the number of characters in input alphabet
#define d 256
/* pat -> pattern
 txt -> text
 q -> A prime number
*/
void search(char *pat, char *txt, int q)
{
 int M = strlen(pat);
 int N = strlen(txt);
 int i, j;
 int p = 0; // hash value for pattern
 int t = 0; // hash value for txt
 int h = 1;
 // The value of h would be "pow(d, M-1)%q"
 for (i = 0; i < M-1; i++)
 h = (h*d)%q;
 // Calculate the hash value of pattern and first window of text
 for (i = 0; i < M; i++)
 {
```

```
 p = (d*p + pat[i])%q;
 t = (d*t + txt[i])%q;
 }
 // Slide the pattern over text one by one
 for (i = 0; i <= N - M; i++)
 {
 // Check the hash values of current window of text and pattern
 // If the hash values match then only check for characters on by
 if ( p == t )
 {
 /* Check for characters one by one */
 for (j = 0; j < M; j++)
 {
 if (txt[i+j] != pat[j])
 break;
 }
 if (j == M) // if p == t and pat[0...M-1] = txt[i, i+1, ...i+M-1]
 {
 printf("Pattern found at index %d \n", i);
 }
 }
 // Calculate hash value for next window of text: Remove leading
int digit;
 // add trailing digit
 if( i < N-M )
 {
 t = (d*(t - txt[i]*h) + txt[i+M])%q;
 // We might get negative value of t, converting it to positive
 if(t < 0)
 t = (t + q);
 }
 }
}
/* Driver program to test above function */
int main()
{
 char *txt = "Hello its okay";
 char *pat = "Hell";
 int q = 101; // A prime number
 search(pat, txt, q);
 return 0;
}
```

Output :-

```
File Edit Selection View Go Run ...        Untitled (Workspace)
Get Started | 6.c | 7.c | queen.c 3 | rabin.c 1
c program > DS_LAB > stack > rabin.c > main()
56  /* Driver program to test above function */
57  int main()
DEBUG CONSOLE  PROBLEMS 4  OUTPUT  TERMINAL                powershell
  ^~
rabin.c: At top level:
rabin.c:64:2: error: expected identifier or '(' before '.' token
 }.
  ^
PS C:\Users\User\Desktop\c program\DS_LAB\stack> gcc rabin.c
rabin.c:64:2: error: expected identifier or '(' before '.' token
 }.
  ^
PS C:\Users\User\Desktop\c program\DS_LAB\stack> gcc rabin.c
PS C:\Users\User\Desktop\c program\DS_LAB\stack> gcc rabin.c -o rabin.exe
PS C:\Users\User\Desktop\c program\DS_LAB\stack> ./rabin
Pattern found at index 0
PS C:\Users\User\Desktop\c program\DS_LAB\stack>
```

KMP Algorithm :-

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
// Function to implement the KMP algorithm
void KMP(const char* text, const char* pattern, int m, int n)
{
 // base case 1: pattern is NULL or empty
 if (*pattern == '\0' || n == 0) {
 printf("The pattern occurs with shift 0");
 }
 // base case 2: text is NULL, or text's length is less than that of pattern's
 if (*text == '\0' || n > m) {
 printf("Pattern not found");
 }
 // next[i] stores the index of the next best partial match
 int next[n + 1];
 for (int i = 0; i < n + 1; i++) {
 next[i] = 0;
 }
 for (int i = 1; i < n; i++)
 {
 int j = next[i + 1];
```

```
 while (j > 0 && pattern[j] != pattern[i]) {
 j = next[j];
 }
 if (j > 0 || pattern[j] == pattern[i]) {
 next[i + 1] = j + 1;
 }
 }
 for (int i = 0, j = 0; i < m; i++)
 {
 if (*(text + i) == *(pattern + j))
 {
 if (++j == n) {
 printf("The pattern occurs with shift %d\n", i - j + 1);
 }
 }
 else if (j > 0)
 {
 j = next[j];
 i--; // since `i` will be incremented in the next iteration
 }
 }
}
// Program to implement the KMP algorithm in C
int main(void)
{
 char* text = "ABCABAABCABAC";
 char* pattern = "CAB";
 int n = strlen(text);
 int m = strlen(pattern);
 KMP(text, pattern, n, m);
 return 0;
}
```

Output :-

```
DEBUG CONSOLE   PROBLEMS 5   OUTPUT   TERMINAL                powershell

Pattern found at index 0
PS C:\Users\User\Desktop\c program\DS_LAB\stack> gcc kmp.c -o kmp.exe
PS C:\Users\User\Desktop\c program\DS_LAB\stack> ./kmp
The pattern occurs with shift 2
The pattern occurs with shift 8
PS C:\Users\User\Desktop\c program\DS_LAB\stack> ./kmp
The pattern occurs with shift 2
The pattern occurs with shift 8
PS C:\Users\User\Desktop\c program\DS_LAB\stack>
```