**Assignment – 8**

**Q1. Implementation of optimal parenthesis of given sequence of matrix using Matric Chain Multiplication and display the intermediate tables.**

#include <limits.h>  
#include <stdio.h>  
#include <stdlib.h>  
  
int\*\* optimal\_matrix\_order(int\*, int);  
void print\_optimal\_parenthesis(int\*\*, int, int);  
void print\_matrix(char[], int\*\*, int);  
  
int main() {  
 int n, i;  
 printf("Enter the no. of matrices: ");  
 scanf("%d", &n);  
 int\* p = (int\*)malloc((n + 1) \* sizeof(int));  
  
 if (!p) {  
 printf("Memory was not allocated\n");  
 exit(0);  
 }  
 printf("Enter the dimensions: ");  
  
 for (i = 0; i <= n; i++) {  
 scanf("%d", &p[i]);  
 }  
 int\*\* s = optimal\_matrix\_order(p, n);  
 print\_matrix("s-matrix:", s, n);  
 printf("The optimal parenthesis: ");  
 print\_optimal\_parenthesis(s, 0, n - 1);  
 printf("\n");  
  
 for (i = 0; i < n; i++) {  
 free(s[i]);  
 }  
 free(s);  
 free(p);  
 return 0;  
}

int\*\* optimal\_matrix\_order(int\* p, int n) {  
 int \*\*m = (int\*\*)malloc(n \* sizeof(int\*)), \*\*s = (int\*\*)malloc(n \* sizeof(int\*)), l, i, j, k, q;  
  
 if (!m || !s) {  
 printf("Memory was not allocated\n");  
 exit(0);  
 }  
 for (i = 0; i < n; i++) {  
 m[i] = (int\*)malloc(n \* sizeof(int));  
 s[i] = (int\*)malloc(n \* sizeof(int));  
  
 if (!m[i] || !s[i]) {  
 printf("Memory was not allocated\n");  
 exit(0);  
 }  
 for (j = 0; j <= i; j++) {  
 m[i][j] = 0;  
 s[i][j] = -1;  
 }  
 }  
 for (l = 1; l < n; l++) {  
 for (i = 0; i < n - l; i++) {  
 j = i + l;  
 m[i][j] = INT\_MAX;  
  
 for (k = i; k < j; k++) {  
 q = m[i][k] + m[k + 1][j] + p[i] \* p[k + 1] \* p[j + 1];  
  
 if (q < m[i][j]) {  
 m[i][j] = q;  
 s[i][j] = k;  
 }  
 }  
 }  
 }  
 print\_matrix("m-matrix:", m, n);  
  
 for (i = 0; i < n; i++) {  
 free(m[i]);  
 }  
 free(m);  
 return s;  
}

void print\_optimal\_parenthesis(int\*\* s, int i, int j) {  
 if (i == j) {  
 printf("A%d", i);  
 } else {  
 printf("(");  
 print\_optimal\_parenthesis(s, i, s[i][j]);  
 print\_optimal\_parenthesis(s, s[i][j] + 1, j);  
 printf(")");  
 }  
}

**Output**:

Enter the no. of matrices: 6  
Enter the dimensions: 5 10 3 12 5 50 6  
  
m-matrix:  
0 150 330 405 1655 2010  
0 0 360 330 2430 1950  
0 0 0 180 930 1770  
0 0 0 0 3000 1860  
0 0 0 0 0 1500  
0 0 0 0 0 0  
  
s-matrix:  
 -1 0 1 1 3 1  
 -1 -1 1 1 1 1  
 -1 -1 -1 2 3 3  
 -1 -1 -1 -1 3 3  
 -1 -1 -1 -1 -1 4  
 -1 -1 -1 -1 -1 -1  
  
The optimal parenthesis: ((A0A1)((A2A3)(A4A5)))