CHAIN MATRIX

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

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

void matrixChain(int *dim, int n)
{
  int i, j, k, l, q;
  int **m = (int **)malloc(n * sizeof(int *));
  int **s = (int **)malloc(n * sizeof(int *));
  for (i = 0; i < n; i++)
  {
    m[i] = (int *)malloc(n * sizeof(int));
    s[i] = (int *)malloc(n * sizeof(int));
  }
  for (i = 1; i < n; i++)
  {
    m[i][i] = 0;
  }
  for (l = 2; l < n; l++)
  {
    for (i = 1; i < n - l + 1; i++)
    {
      j = i + l - 1;
      m[i][j] = INT_MAX;
      for (k = i; k <= j - 1; k++)
      {
        q = m[i][k] + m[k + 1][j] + dim[i - 1] * dim[k] * dim[j];
        if (q < m[i][j])
        {
          m[i][j] = q;
          s[i][j] = k;
        }
      }
    }
  }
  printf("The m table is: \n");
  for (i = 1; i < n; i++)
  {
    for (j = 1; j < n; j++)
    {
      printf("%d ", m[i][j]);
    }
    printf("\n");
  }
  printf("The s table is: \n");
  for (i = 1; i < n; i++)
  {
    for (j = 1; j < n; j++)
    {
      printf("%d ", s[i][j]);
    }
    printf("\n");
  }

  printf("The minimum multiplication is: %d\n", m[1][n - 1]);
  printf("Parenthesization is: ");
  parenPrint(s, 1, n - 1);
  printf("\n");
}

int main()
{
  int n, i;
  printf("Enter the number of matrices: ");
  scanf("%d", &n);
  int *dim = (int *)malloc((n + 1) * sizeof(int));
  printf("Enter the dimensions of the matrices: ");
  for (i = 0; i <= n; i++)
  {
    scanf("%d", &dim[i]);
  }

  matrixChain(dim, n + 1);

  return 0;
}
```

LCS

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define max(a, b) (a > b ? a : b)

void printLCS(char *x, char *y, int n, int m, int **dp)
{
  int l = dp[n][m];
  char *s = (char *)malloc((l + 1) * sizeof(char));
  s[l] = '\0';
  int i = n, j = m;
  while (i > 0 && j > 0)
  {
    if (x[i - 1] == y[j - 1])
    {
      s[l - 1] = x[i - 1];
      l--;
      i--;
      j--;
    }
    else if (dp[i - 1][j] > dp[i][j - 1])
    {
      i--;
    }
    else
    {
      j--;
    }
  }
  printf("The longest common subsequence is: %s\n", s);
  free(s);
}

void lcsDP(char *x, char *y, int n, int m)
{
  int i, j;
  int **dp = (int **)malloc((n + 1) * sizeof(int *));
  for (i = 0; i <= n; i++)
  {
    dp[i] = (int *)malloc((m + 1) * sizeof(int));
  }
  for (i = 0; i <= n; i++)
  {
    dp[i][0] = 0;
  }
  for (i = 0; i <= m; i++)
  {
    dp[0][i] = 0;
  }
  for (i = 1; i <= n; i++)
  {
    for (j = 1; j <= m; j++)
    {
      if (x[i - 1] == y[j - 1])
      {
        dp[i][j] = dp[i - 1][j - 1] + 1;
      }
      else
      {
        dp[i][j] = max(dp[i - 1][j], dp[i][j - 1]);
      }
    }
  }
  printf("The dp table is: \n");
  for (i = 0; i <= n; i++)
  {
    for (j = 0; j <= m; j++)
    {
      printf("%d ", dp[i][j]);
    }
    printf("\n");
  }
  printf("The length of the longest common subsequence is: %d\n",
dp[n][m]);
  printLCS(x, y, n, m, dp);
  for (int i = 0; i <= n; i++)
  {
    free(dp[i]);
  }
  free(dp);
}

int main()
{
  int n, m;
  char x[20], y[20];
  printf("Enter the first string: ");
  scanf("%s", x);
  printf("Enter the second string: ");
  scanf("%s", y);
  n = strlen(x);
  m = strlen(y);
  lcsDP(x, y, n, m);
  return 0;
}
```