|  |  |
| --- | --- |
| **Name** | Ritik Singh |
| **UID no.** | 2021700061 |
| **Experiment No.** | 5 |

|  |  |
| --- | --- |
| **AIM:** | Implementation of Matrix Chain Multiplication |
| **Program 1** | |
| **PROBLEM STATEMENT :** | Implementation of Matrix Chain Multiplication |
| **ALGORITHM/**  **THEORY:** | 1. n length[p]-1  2. for i ← 1 to n  3. do m [i, i] ← 0  4. for l ← 2 to n // l is the chain length  5. do for i ← 1 to n-l + 1  6. do j ← i+ l -1  7. m[i,j] ← ∞  8. for k ← i to j-1  9. do q ← m [i, k] + m [k + 1, j] + pi-1 pk pj  10. If q < m [i,j]  11. then m [i,j] ← q  12. s [i,j] ← k  13. return m and s. |
| **PROGRAM:** | #include <stdio.h>  #include<stdlib.h>  Bracketing(int\* N, int a, int b, int n){      int m=\*(N+a\*n+b);      if(m-a<=1 && b-m-1>1){          if(m-a==0){              printf("M%d\*(",a+1);              Bracketing(N,m+1,b,n);              printf(")");          }          else if(m-a==1){              printf("(M%d\*M%d)(",a+1,m+1);              Bracketing(N,m+1,b,n);              printf(")");          }      }      else if(m-a>1 && b-m-1<=1){          if(b-m-1==0){              printf("(");              Bracketing(N,a,m,n);              printf(")\*M%d",b+1);          }          else if(b-m-1==1){              printf("(",a,m);              Bracketing(N,a,m,n);              printf(")(M%d\*M%d)",m+2,b+1);          }      }      else if(m-a<=1 && b-m-1<=1){          if(m-a==1 && b-m-1==0)              printf("(M%d\*M%d)\*M%d",a+1,m+1,m+2);          else if(m-a==0 && b-m-1==1)              printf("M%d\*(M%d\*M%d)",a+1,m+2,b+1);          else if(m-a==1 && b-m-1==1)              printf("(M%d\*M%d)(M%d\*M%d)",a+1,m+1,m+2,b+1);      }      else{          printf("(");          Bracketing(N,a,m,n);          printf(")(");          Bracketing(N,m+1,b,n);          printf(")");      }  }  Chain(int\* A, int\* M, int \*N, int m, int n){      int i,l;      for(i=0;i<m;i++){          printf("\n  }\n\nM[%d,%d]= min of\n  {\n        M[%d,%d] + M[%d,%d] + (d%d\*d%d\*d%d) = ",i+1,n-m+i+1,i+1,i+1,i+2,n-m+i+1,i+1,i+2,n-m+i+2);          printf("%d + %d + (%d\*%d\*%d) = ",\*(M+i\*n+i),\*(M+(i+1)\*n+n-m+i),\*(A+i),\*(A+i+1),\*(A+n-m+i+1));          \*(M+n\*i+n-m+i)=\*(M+i\*n+i)+\*(M+(i+1)\*n+n-m+i)+(\*(A+i))\*(\*(A+i+1))\*(\*(A+n-m+i+1));          \*(N+n\*i+n-m+i)=i;          printf("%d",\*(M+n\*i+n-m+i));          for(l=i+1;l<=n-m+i-1;l++){              if(\*(M+n\*i+n-m+i)>(\*(M+i\*n+l)+\*(M+(l+1)\*n+n-m+i)+(\*(A+i))\*(\*(A+l+1))\*(\*(A+n-m+i+1)))){                  printf("\n        M[%d,%d] + M[%d,%d] + (d%d\*d%d\*d%d) = ",i+1,l+1,l+2,n-m+i+1,i+1,l+2,n-m+i+2);                  printf("%d + %d + (%d\*%d\*%d) = ",\*(M+i\*n+l),\*(M+(l+1)\*n+n-m+i),\*(A+i),\*(A+l+1),\*(A+n-m+i+1));                  \*(M+n\*i+n-m+i)=(\*(M+i\*n+l)+\*(M+(l+1)\*n+n-m+i)+(\*(A+i))\*(\*(A+l+1))\*(\*(A+n-m+i+1)));                  \*(N+n\*i+n-m+i)=l;                  printf("%d",\*(M+n\*i+n-m+i));              }              else{                  printf("\n        M[%d,%d] + M[%d,%d] + (d%d\*d%d\*d%d) = ",i+1,l+1,l+2,n-m+i+1,i+1,l+2,n-m+i+2);                  printf("%d + %d + (%d\*%d\*%d) = %d",\*(M+i\*n+l),\*(M+(l+1)\*n+n-m+i),\*(A+i),\*(A+l+1),\*(A+n-m+i+1),(\*(M+i\*n+l)+\*(M+(l+1)\*n+n-m+i)+(\*(A+i))\*(\*(A+l+1))\*(\*(A+n-m+i+1))));              }          }      }      if(m>=2)          Chain(A,M,N,m-1,n);  }  main()  {      int  i,j,k,l,n;      printf("Please enter the total number of matrix dimension sequence, you wanna multiply:");      scanf("%d",&n);      printf("\nNow enter all the order of matrices maintaining a proper sequence:");      int A[n],M[n-1][n-1],N[n-1][n-1];      for(i=0;i<n;i++)      {          // scanf("%d",&A[i]);          A[i] = (rand() % (46 - 15 + 1) + 15);      }      for(i=0;i<n-1;i++)          for(j=0;j<n-1;j++){              M[i][j]=0;              N[i][j]=0;          }      Chain(A,M,N,n-2,n-1);      printf("\n  }\n\nThe matrix having minimum number of multiplication is:\n");      for(i=0;i<n-1;i++){          printf("\n\n");          for(j=0;j<n-1;j++){              if(i>j)                  printf("\t-");              else                  printf("\t%d",M[i][j]);          }      }      printf("\n\nSo the minimum number of multiplication required is: %d.",M[0][n-2]);      printf("\n\nThe parenthesization would be like:\t");      Bracketing(N,0,n-2,n-1);      printf("\n\n\n");  } |
| **RESULT:** | |
| **CONCLUSION:** | Implemented and Learned the Matrix Chain Multiplication Algorithm. |