STOCK MARKET PROBLEM

PROJECT REPORT

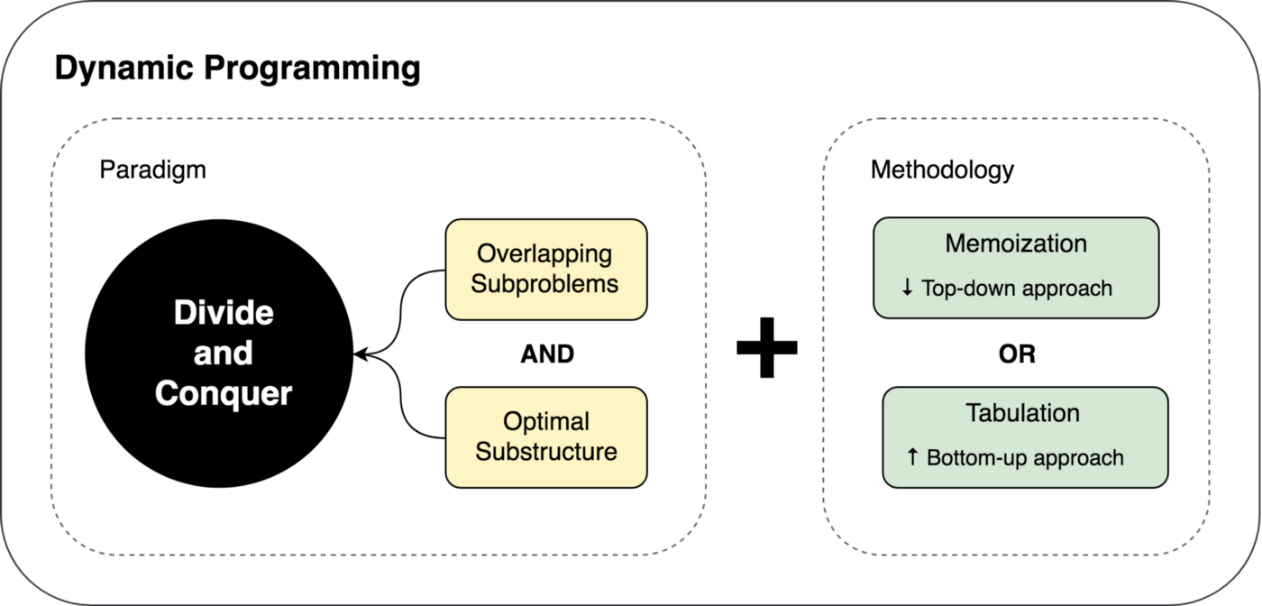
BY

B.SURESH KUMAR-1602-20-737-051

J.BHASKAR-1602-20-737-305

INTRODUCTION:

This problem is solved using Dynamic programming. Dynamic Programming is a technique in computer programming that helps to efficiently solve a class of problems that have overlapping subproblems and optimal substructure property. The idea is to simply store the results of subproblems, so that we do not have to re-compute them when needed later. This simple optimization reduces time complexities from exponential to polynomial.



ABSTRACT:



SCOPE: 

PROJECT DESCRIPTION:

* The main function takes input for the stock values.
* Then, these values are passed to the function maxProfit as parameters.
* In the maxProfit function, we created a matrix to memoize values.
* The final result is returned by this function which is displayed by the standard output.

Input:

First line consists of no.of stock values and second line consists of values of stock on each day

Output:

Prints the Maximum profit that could be made me buying.

CODE:  
#include<bits/stdc++.h>

using namespace std;

int maxProfit(int n, vector<int> &a)

{

int i;

int mx;

int mn;

int profit;

vector<vector<int> >dp(2,vector<int>(n));

//dp[0][i]=maximum profit by buying and selling a stock just once a day by using only values a[0...i]

//dp[1][i]=maximum profit by buying and selling a stock just once a day by using only values a[i...n-1]

//i is the value at which stock will be sold

dp[0][0]=0;

mn=a[0];

for(i=1;i<n;i++)

{

dp[0][i]=max(dp[0][i-1],a[i]-mn);

if(a[i]<mn)

{

mn=a[i];

}

}

//i is the value at which stock will be purchased

dp[1][n-1]=0;

mx=a[n-1];

profit=dp[0][n-1];

for(i=n-2;i>=2;i--)

{

dp[1][i]=max(dp[1][i+1],mx-a[i]);

profit=max(profit,dp[1][i]+dp[0][i-1]);

if(a[i]>mx)

{

mx=a[i];

}

}

return profit;

}

int main()

{

int i,n;

cout<<"Enter the number of stock values ";

cin>>n;

vector<int> a(n);

cout<<"Enter the values of stock on each day "<<endl;

for(i=0;i<n;i++)

cin>>a[i];

cout<<"Maximum profit that could be made by buying and selling a share at most twice"<<endl;

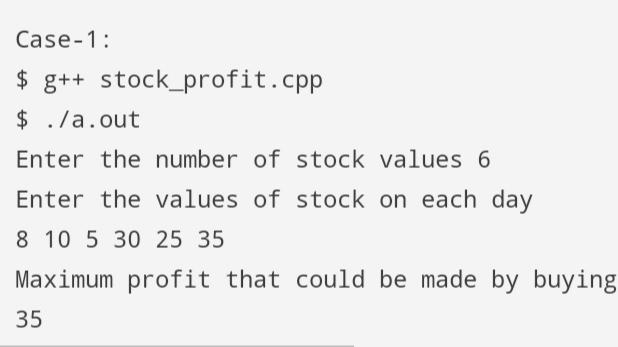
cout<<maxProfit(n,a);

cout<<endl;

return 0;

}

OUTPUTS:



CONCLUSION& FUTURE SCOPE:

After doing this project we conclude that this Stock market problem is very much useful for further use. We can easily calculate the profits for stock markets and we slso know that

Dynamic programming made it easier to solve the ptoblem.

Apart from regular curriculum what you have learned more?

We learnt how to solve different types of problems apart from the standard ones like knapsack, TSP, e.t.c by using dynamic programming approach.