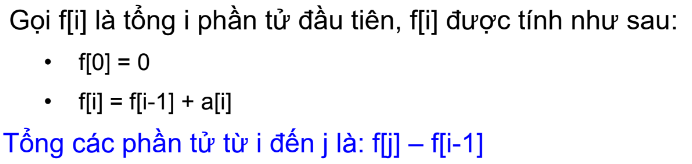
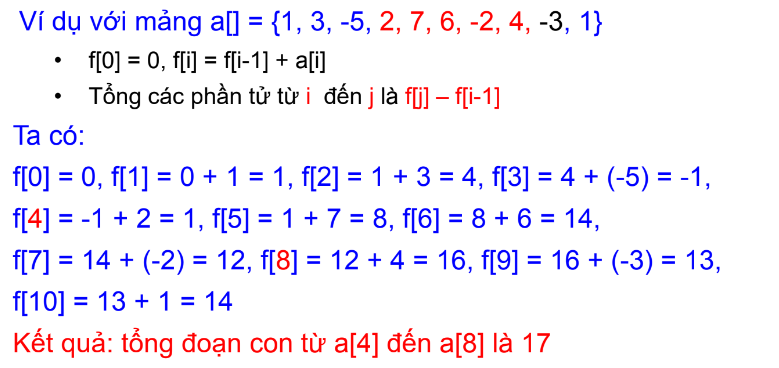
# CHIA ĐỂ TRỊ

## **Bài 1: Tìm tổng lớn nhất của đoạn con liên tục.**

\*Ý tưởng





\*CODE

|  |
| --- |
| #include<iostream>  using namespace std;  int main() {  // int n;  // cin>>n;  // int a[=];  // for(int i=0;i<n;i++)  // cin>>a[i];  int a[] = {-2, -3, 4, -1, -2, 1, 5, -3};  int n = sizeof(a)/sizeof(a[0]);  int sum=0,maxSum=INT\_MIN,start,end,s=0;  for(int i=0; i<n; i++) {  sum+=a[i];  if(maxSum<sum) {  maxSum=sum;  start=s;  end=i;  }  if(sum<0) {  sum=0;  s=s+1;  }  }  cout<<"Max sum of subarray: "<<maxSum;  cout<<"\nStart: "<<start;  cout<<"\nEnd: "<<end;  } |

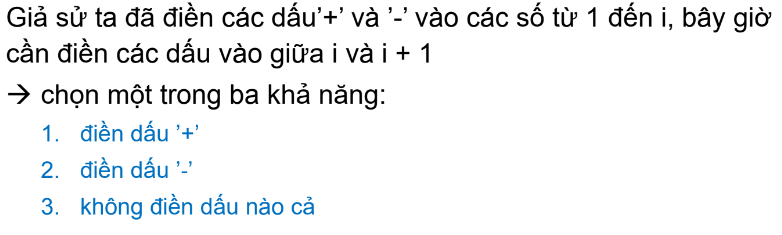
## **Bài 2: Phân tích thiết kế đánh giá độ phức tạp của thuật toán sắp xếp nhanh (Quick sort).**

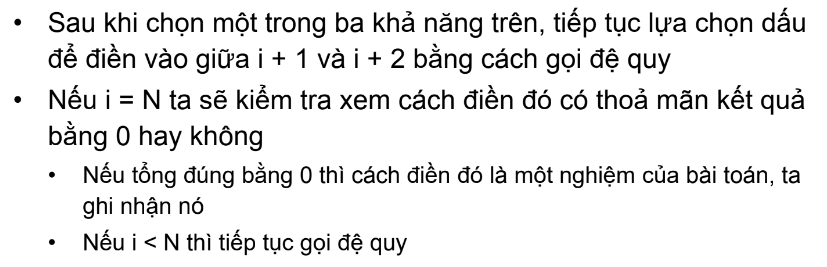
|  |
| --- |
| Là thuật toán sắp xếp dựa trên kỹ thuật chia để trị  • Các bước:  • Cơ sở: dãy chỉ còn không quá 1 phần tử thì dãy đó đã được sắp xếp => không cần thao tác thêm gì cả  • Chia: gồm hai công việc  • Chọn một phần tử trong dãy làm phần tử chốt p (pivot)  • Phân hoạch: chia dãy đã cho thành hai dãy con, dãy con trái (L) sẽ gồm những phần tử nhỏ hơn hoặc bằng phần tử chốt và dãy con phải (R) là những phần tử lớn hơn phần tử chốt  • Trị: lặp lại một cách đệ quy thuật toán với 2 dãy con trái và phải  • Tổng hợp: dãy được sắp xếp L,p,R  - Với thuật toán Quick Sort thì việc chọn phần tử chốt có vai trò quyết định đối với hiệu quả của thuật toán. Các cách chọn phần tử chốt:  • Chọn phần tử đầu tiên  • Chọn phần tử cuối cùng  • Chọn phần tử đúng giữa dãy  • Chọn phần tử trung vị trong 3 phần tử đứng đầu, đứng cuối và đứng giữa  • Chọn phần tử ngẫu nhiên  \*CODE   1. **//void quickSort(int a[],int low,int high){** 2. **// int i=low,j=high;** 3. **// int pivot=a[(low+high)/2];** 4. **// while(i<=j){** 5. **// while(a[i]<pivot)** 6. **// i++;** 7. **// while(a[j]>pivot)** 8. **// j--;** 9. **// if(i<=j){** 10. **// swap(a[i],a[j]);** 11. **// i++;** 12. **// j--;** 13. **// }** 14. **// }** 15. **// if(low<j)** 16. **// quickSort(a,low,j);** 17. **// if(i<high)** 18. **// quickSort(a,i,high);** 19. **//}** |

# QUAY LUI

## **Bài 1: Biểu thức zero. Cho một số tự nhiên N ≤ 9. Giữa các số từ 1 đến N hãy thêm vào các dấu + và - sao cho kết quả thu được bằng 0. Hãy viết chương trình tìm tất cả các khả năng có thể. Áp dụng thuật toán đệ quy quay lui để giải quyết bài toán này.**

\*Ý tưởng



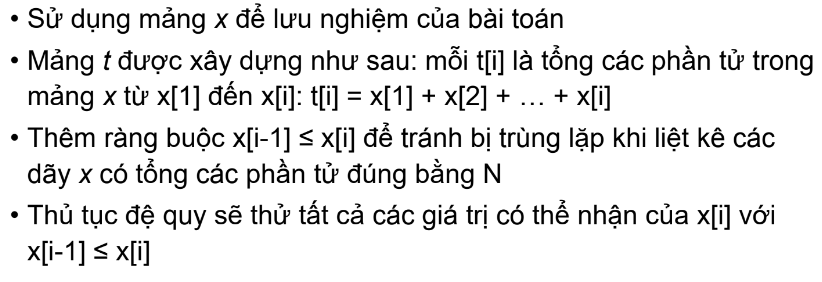


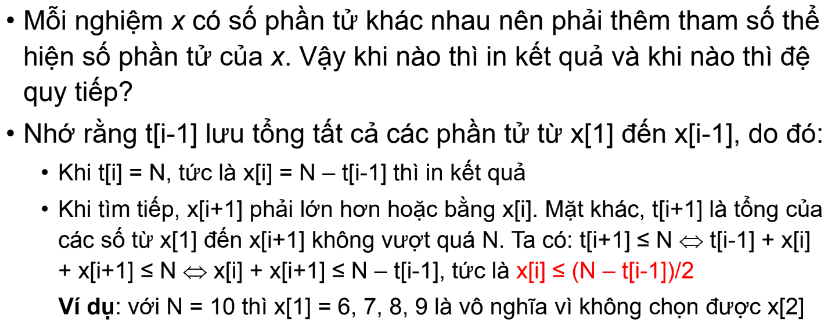
\*CODE

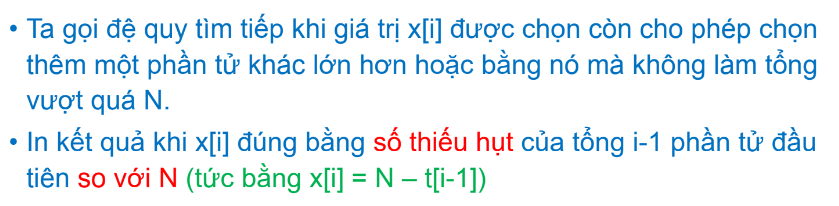
|  |
| --- |
| 1. **#include <bits/stdc++.h>** 2. **using namespace std;** 3. **#define ll long long** 4. **#define fast\_cin ios\_base::sync\_with\_stdio(false);cin.tie(NULL);cout.tie(NULL)** 5. **#define endl "\n"** 6. **#define ll long long** 7. **#define For(i, a, b) for (ll i = a; i <= b; ++i)** 8. **#define Fo(i, a, b) for (ll i = a; i < b; ++i)** 9. **#define Ford(i, a, b) for (ll i = a; i >= b; --i)** 10. **#define fi first** 11. **#define se second** 13. **int check;** 15. **int precedence(char c){** 16. **if (c == '+'|| c == '-') return 1;** 17. **if (c == '\*'|| c == '/') return 2;** 18. **return 0;** 19. **}** 21. **ll applyOp(ll a, ll b, char c){** 22. **switch(c) {** 23. **case '+': return a + b;** 24. **case '-': return a - b;** 25. **case '\*': return a \* b;** 26. **case '/': return a / b;** 27. **}** 28. **}** 30. **ll evaluate(string tokens) {** 31. **ll i;** 33. **stack <ll> values;** 34. **stack <char> ops;** 36. **Fo(i, 0, tokens.length()) {** 37. **if (tokens[i] == '(') {** 38. **ops.push(tokens[i]);** 39. **} else if (isdigit(tokens[i])) {** 40. **ll val = 0;** 41. **while (i < tokens.length() && isdigit(tokens[i])) {** 42. **val = (val\*10) + (tokens[i]-'0');** 43. **i++;** 44. **}** 46. **values.push(val);** 47. **i--;** 48. **} else if (tokens[i] == ')') {** 49. **while (!ops.empty() && ops.top() != '(') {** 50. **ll val2 = values.top();** 51. **values.pop();** 53. **ll val1 = values.top();** 54. **values.pop();** 56. **char op = ops.top();** 57. **ops.pop();** 59. **values.push(applyOp(val1, val2, op));** 60. **}** 61. **if (!ops.empty()) ops.pop();** 62. **} else {** 63. **while (!ops.empty() && precedence(ops.top()) >= precedence(tokens[i])) {** 64. **ll val2 = values.top();** 65. **values.pop();** 67. **ll val1 = values.top();** 68. **values.pop();** 70. **ll op = ops.top();** 71. **ops.pop();** 73. **values.push(applyOp(val1, val2, op));** 74. **}** 76. **ops.push(tokens[i]);** 77. **}** 78. **}** 80. **while (!ops.empty()) {** 81. **ll val2 = values.top();** 82. **values.pop();** 84. **ll val1 = values.top();** 85. **values.pop();** 87. **char op = ops.top();** 88. **ops.pop();** 90. **values.push(applyOp(val1, val2, op));** 91. **}** 92. **return values.top();** 93. **}** 95. **void printVal(string s) {** 96. **Fo(i, 0, s.size()) {** 97. **cout << s[i];** 98. **}** 99. **cout << "=0" << endl;** 100. **}** 102. **void backTracking(ll i, string s, ll \*val, ll n) {** 103. **if (i == n) {** 104. **s += to\_string(val[i]);** 105. **if(evaluate(s) == 0) {** 106. **printVal(s);** 107. **check = 1;** 108. **}** 109. **return;** 110. **}** 112. **backTracking(i + 1, s + to\_string(val[i]) + '+', val, n);** 113. **backTracking(i + 1, s + to\_string(val[i]) + '-', val, n);** 114. **backTracking(i + 1, s + to\_string(val[i]), val, n);** 115. **}** 117. **int main() {** 118. **ll n;** 119. **ll val[16];** 120. **cin >> n;** 121. **string s = "";** 122. **For(i, 0, 15) {** 123. **val[i] = i;** 124. **}** 125. **backTracking(1, s, val, n);** 126. **if (!check) {** 127. **cout << "No such answer !!";** 128. **}** 130. **return 0;** 131. **}** |

## **Bài 2: Cho một tự nhiên N ≤ 30. Tìm tất cả các cách phân tích số N thành tổng của các số nguyên dương. Các cách phân tích là hoán vị của nhau thì chỉ tính là một cách.**

*\*Ý tưởng*







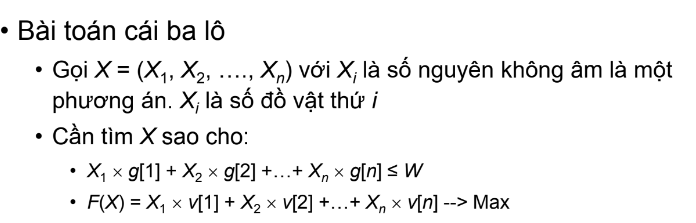
\*CODE

|  |
| --- |
| 1. **#include<bits/stdc++.h>** 2. **using namespace std;** 3. **void printValue(int k,int x[],int n){** 4. **cout<<endl<<n<<" = ";** 5. **for(int i=1;i<k;i++){** 6. **cout<<x[i]<<" + ";** 7. **}** 8. **cout<<x[k];** 9. **}** 10. **void backTracking(int i,int x[],int t[],int n){** 11. **for(int j=x[i-1];j<=(n-t[i-1])/2;j++){** 12. **x[i]=j;** 13. **t[i]=t[i-1]+j;** 14. **backTracking(i+1,x,t,n);** 15. **}** 16. **x[i]=n-t[i-1];** 17. **printValue(i,x,n);** 18. **}** 19. **int main(){** 20. **int n,x[35],t[35];** 21. **cin>>n;** 22. **x[0]=1;** 23. **t[0]=0;** 24. **backTracking(1,x,t,n);** 25. **}** |

**Bài 3: Bài toán cái ba lô**

**Cho một cái ba lô có thể đựng một trọng lượng W và n loại đồ vật, mỗi đồ vật i có một trọng lượng g[i] và một giá trị v[i]. Tất cả các loại đồ vật đều có số lượng không hạn chế. Tìm một cách lựa chọn các đồ vật đựng vào ba lô, chọn các loại đồ vật nào, mỗi loại lấy bao** **nhiêu sao cho tổng trọng lượng không vượt quá W và tổng giá trị là lớn nhất**

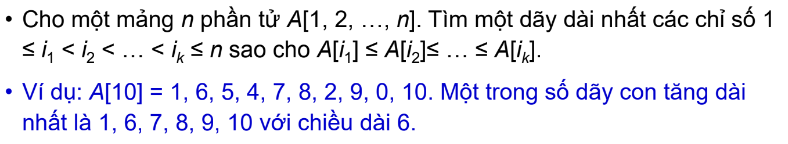
*\*Ý tưởng*

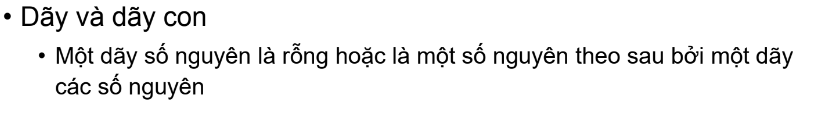


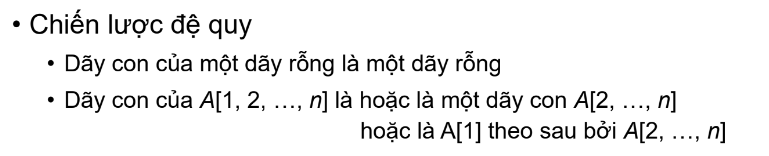
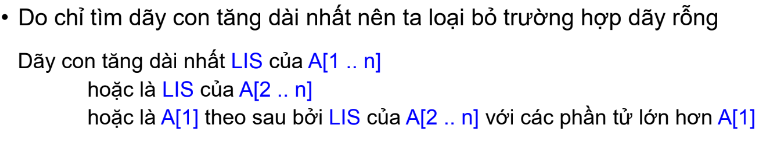
#include "stdafx.h"  
#include<iostream>  
using namespace std;  
  
int m,n,tong,maxd,giatri;  
typedef struct balo  
{  
int kichthuoc;  
int giatri;  
  
};  
balo Balo[100];  
void input()  
{  
cout<<"Nhap kich thuoc cua cai tui: ";  
cin>>m;  
cout<<"Nhap so luong do vat: ";  
cin>>n;  
for(int i=0;i<n;i++)  
{  
cout<<"nhap kich thuoc vat thu "<<i<<" :";  
cin>>Balo[i].kichthuoc;  
cout<<"nhap gia tri vat thu "<<i<<" :";  
cin>>Balo[i].giatri;  
}  
}  
void duyet(int v)  
{  
for(int i = 0;i < n;i ++)  
{  
if(tong+Balo[i].kichthuoc <= m)  
{  
tong = tong + Balo[i].kichthuoc;  
giatri = giatri + Balo[i].giatri;  
if(giatri > maxd)  
maxd = giatri;  
duyet(v+1);  
tong = tong - Balo[i].kichthuoc;  
giatri = giatri - Balo[i].giatri;  
  
}  
}  
}  
int main()  
{  
tong = 0;  
giatri = 0;  
maxd = 0;  
input();  
duyet(1);  
printf("\nGia tri lon nhat co the cam theo la: %d",maxd);  
return 0;  
}

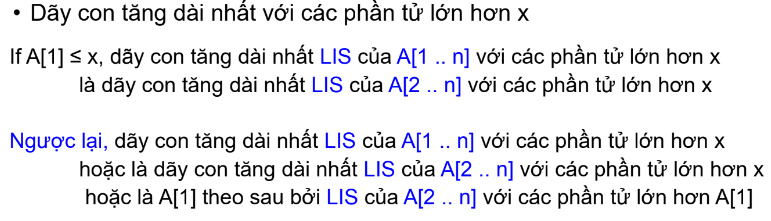
**Bài 4: Dãy con tăng dài nhất**

*\*Ý tưởng*







#include <stdio.h>

#include <algorithm>

using namespace std;

#define nmax=1000

int n;

int a[1000], f[1000];

int main() {

scanf("%d",&n);

int i,j;

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

scanf("%d",&a[i]);

f[i]=1;

}

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

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

if (a[i]<a[j]) f[j]=max(f[i]+1,f[j]);

}

}

int res=0;

for (i=1; i<=n; i++) res=max(res,f[i]);

printf("%d",res);

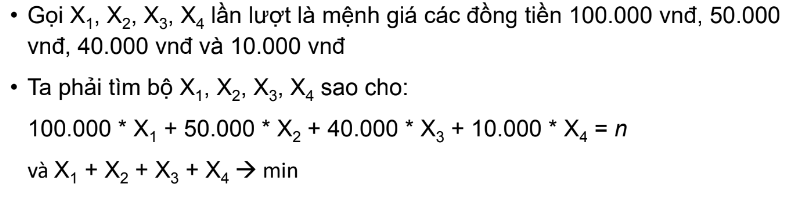
return 0;

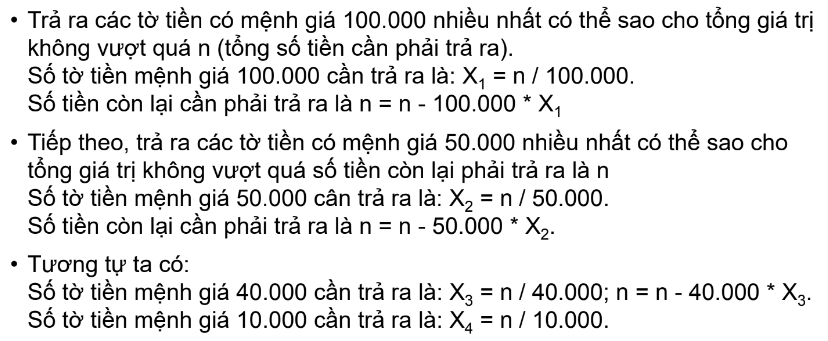
}

# THAM LAM

## **Bài 1. Máy rút tiền ATM. Giả sử có các loại tiền: 100.000 vnđ, 50.000 vnđ, 40.000 vnđ và 10.000 vnđ. Mỗi loại tiền đều có số lượng không hạn chế. Khách hàng cần rút một số tiền n vnđ (tính chẵn đến 10.000 vnđ, hay n chia hết cho 10.000). Phương án trả tiền sao cho trả đủ n vnđ và số tờ tiền phải trả là ít nhất**

*\*Ý tưởng*





|  |
| --- |
| 1. **#include<bits/stdc++.h>** 2. **using namespace std;** 3. **int main()** 4. **{** 5. **int n,a[4]={100,50,40,10};** 6. **cout<<"Nhap so tien can doi: ";** 7. **cin>>n;** 8. **int result=0;** 9. **for(int i=0;i<4;i++)** 10. **{** 11. **result+=n/a[i];** 12. **n%=a[i];** 13. **}** 14. **cout<<"So to tien it nhat: ";** 15. **cout<<result;** 16. **}** |

## **Bài 2: Có n cuộc họp, cuộc họp thứ i bắt đầu vào thời điểm Si và kết thúc ở thời điểm Fi. Do chỉ có một phòng hội thảo nên hai cuộc họp bất kì sẽ được cùng bố trí phục vụ nếu khoảng thời gian làm việc của chúng chỉ giao nhau tại đầu mút. Hãy bố trí phòng họp để phục vụ được nhiều cuộc họp nhất.**

\*CODE

|  |
| --- |
| struct meeting  {  int start;  int index;  int end;  };  bool cmp(meeting a, meeting b)  {  return a.end < b.end;  }  int main()  {  int n;  cout << "Nhap n cuoc hop: ";  cin >> n;  meeting a[n];  for (int i = 0; i < n; i++)  {  cout << "Nhap gio bat dau va ket thuc cuoc hop " << i + 1 << " : ";  a[i].index = i + 1;  cin >> a[i].start;  cin >> a[i].end;  }  sort(a, a + n, cmp);  cout << "\nSau khi sap xep cuoc hop\n";  for (meeting x : a)  {  cout << x.index << " " << x.start << " " << x.end << endl;  }  int dem = 0;  meeting indexToSave = a[0];  vector<int> v;  for (int i = 1; i < n; i++)  {  if (indexToSave.end <= a[i].start)  {  dem++;  indexToSave = a[i];  v.push\_back(a[i].index);  }  }  cout << "\nSo cuoc hop co the tham gia toi da la: " << dem + 1 << endl;  cout << "\nCac buoc thuc hien la: " << a[0].index << " ";  for (int x : v)  cout << x << " ";  } |

## 

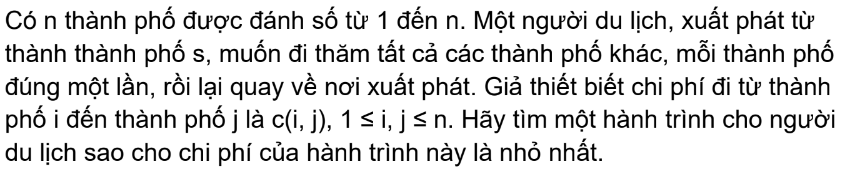
## **Bài 3: Trung tâm tính toán hiệu năng cao nhận được đơn đặt hàng của n khách hàng. Khách hàng i muốn sử dụng máy trong khoảng thời gian từ ai đến bi và trả tiền thuê là ci. Hãy bố trí lịch thuê máy để tổng số tiền thu được là lớn nhất mà thời gian sử dụng máy của hai khách hàng bất kì được phục vụ đều không giao nhau (cả trung tâm chỉ có một máy cho thuê)**

\*CODE

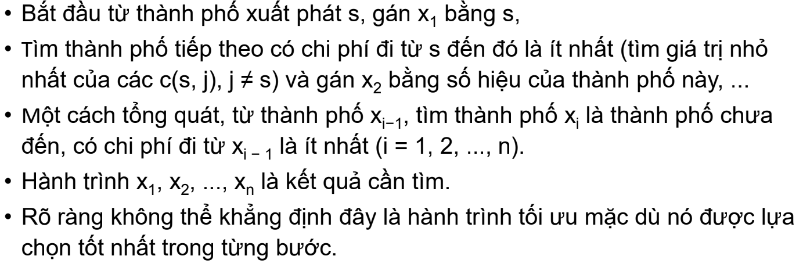
|  |
| --- |
| struct rentphone  {  int start;  int end;  int profit;  int index;  };  bool cmp(rentphone a, rentphone b){  if(a.start==b.start)  return a.profit>b.profit;  return a.end<b.end;  }  int main()  {  cout<<"Nhap n khach hang: ";  int n;  cin>>n;  rentphone a[n];  for(int i=0;i<n;i++)  {  a[i].index = i+1;  cout<<"Nhap thoi gian bat dau,ket thuc, tien thue: ";  cin>>a[i].start>>a[i].end>>a[i].profit;  }  sort(a,a+n,cmp);  cout<<"\nSau khi sap xep theo thoi gian ket thuc la: \n";  for(rentphone x: a)  cout<<x.index<<" "<<x.start<<" "<<x.end<<" "<<x.profit<<endl;  rentphone indexToSave = a[0];  int resProfit=0;  vector<rentphone> v;  for (int i = 1; i < n; i++)  {  if (indexToSave.end <= a[i].start)  {  resProfit+=a[i].profit;  indexToSave = a[i];  v.push\_back(a[i]);  }  }  cout<<"\nSo tien thu dc sau khi thue la: "<<resProfit+a[0].profit;  cout<<"\n\nCac buoc thuc hien khi cho thue la: "<<endl;  cout<<a[0].index<<" "<<a[0].start<<" "<<a[0].end<<" "<<a[0].profit<<endl;  for(rentphone x: v)  cout<<x.index<<" "<<x.start<<" "<<x.end<<" "<<x.profit<<endl;  } |

# 

# **Bài 4: Người đi du lịch**



*\*Ý tưởng*



#include <bits/stdc++.h>

using namespace std;

#define Fast\_cin ios\_base::sync\_with\_stdio(false);cin.tie(NULL);cout.tie(NULL);

map<int, int> visitedRouteList;

void findMinRoute(int tsp[4][4], int n)

{

int sum = 0;

int counter = 0;

int j = 0, i = 0;

int min = INT\_MAX;

visitedRouteList[0] = 1;

int route[n];

while (i < n && j < n)

{

if (counter >= n - 1) {

break;

}

if (j != i && (visitedRouteList[j] == 0)) {

if (tsp[i][j] < min) {

min = tsp[i][j];

route[counter] = j + 1;

}

}

j++;

if (j == n) {

sum += min;

min = INT\_MAX;

visitedRouteList[route[counter] - 1] = 1;

j = 0;

i = route[counter] - 1;

counter++;

}

}

i = route[counter - 1] - 1;

for (j = 0; j < n; j++) {

if ((i != j) && tsp[i][j] < min) {

min = tsp[i][j];

route[counter] = j + 1;

}

}

sum += min;

cout << sum;

}

int main() {

int n = 4;

int tsp[4][4] = { { -1, 10, 15, 20 },

{ 10, -1, 35, 25 },

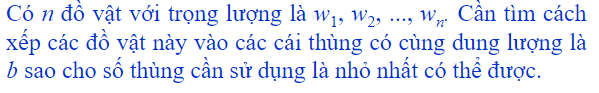
{ 15, 35, -1, 30 },

{ 20, 25, 30, -1 } };

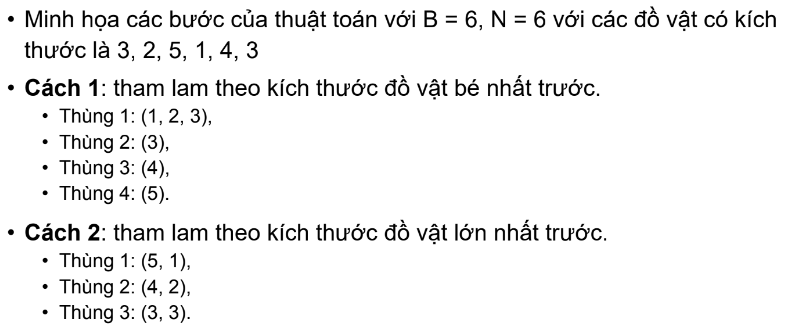
findMinRoute(tsp, 4);

}

**Bài 5: Bài toán đóng thùng**



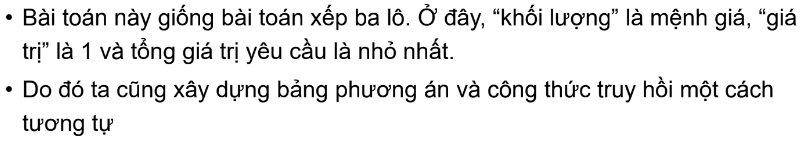
*\*Ý tưởng*

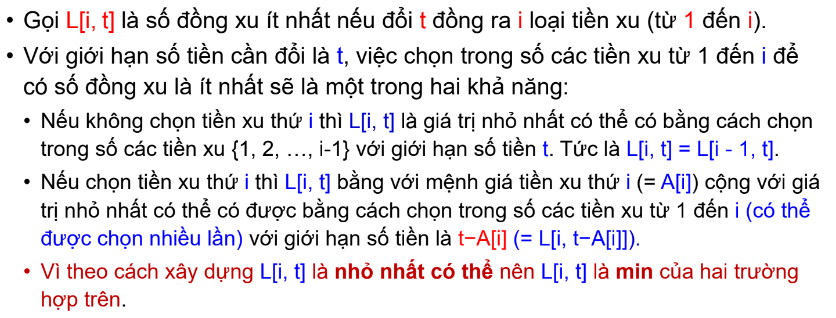


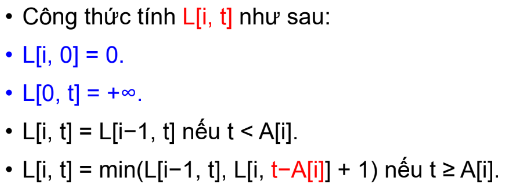
# QUY HOẠCH ĐỘNG

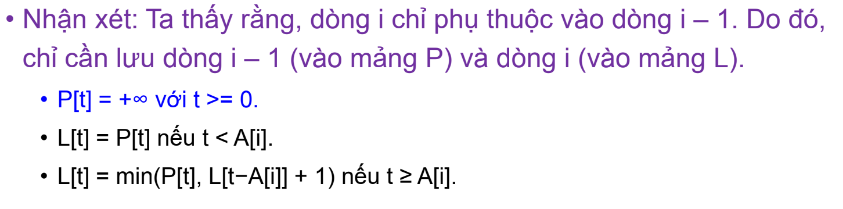
## **Bài 1: Đổi tiền. Ở đất nước Omega người ta chỉ tiêu tiền xu. Có N loại tiền xu, loại thứ i có mệnh giá là Ai đồng. Một người khách du lịch đến Omega du lịch với số tiền M đồng. Ông ta muốn đổi số tiền đó ra tiền xu Omega để tiện tiêu dùng. Ông ta cũng muốn số đồng tiền đổi được là ít nhất (cho túi tiền đỡ nặng khi đi đây đi đó). Bạn hãy giúp ông ta tìm cách đổi tiền**

*\*Ý tưởng*







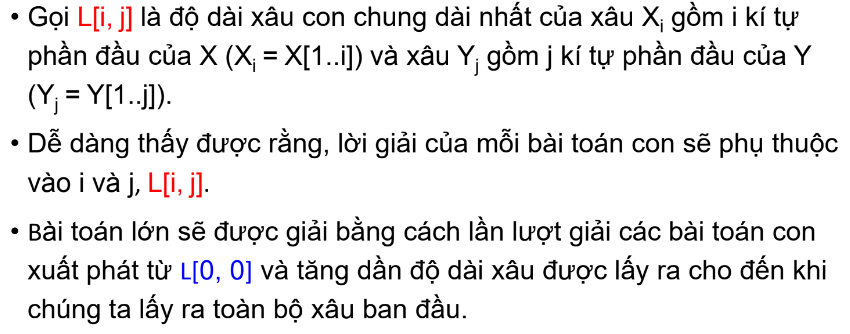


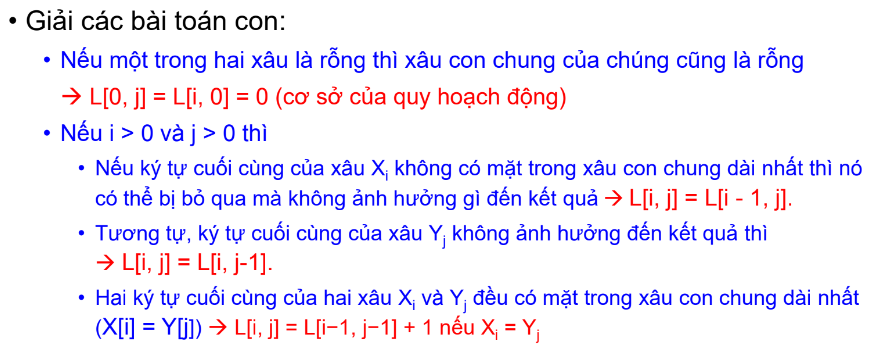
\*CODE

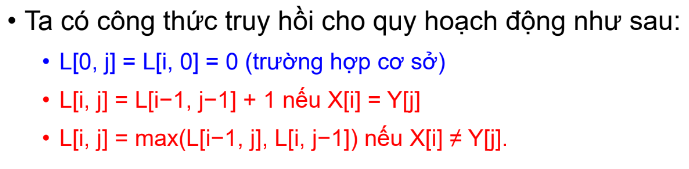
|  |
| --- |
| **#include<bits/stdc++.h>**  **using namespace std;**  **int main(){**  **cout<<"Money need to change: ";**  **int money,n;**  **cin>>money;**  **cout<<"Number of coin: ";**  **cin>>n;**  **int coin[n];**  **cout<<"Coins: ";**  **for(int i=0;i<n;i++){**  **cin>>coin[i];**  **}**  **int dp[money+1];**  **int sol[money+1];**  **dp[0]=0;**  **sol[0]=-1;**  **for(int i=1;i<=money;i++){**  **dp[i]=INT\_MAX-1;**  **sol[i]=-1;**  **}**  **for(int i=0;i<n;i++)**  **for(int j=1;j<=money;j++)**  **if(j>=coin[i]&&dp[j-coin[i]]+1<dp[j]){**  **dp[j]=dp[j-coin[i]]+1;**  **sol[j]=i;**  **}**  **cout<<"Minimum of coins is: "<<dp[money];**  **int start = sizeof(sol)/sizeof(int)-1;**  **if(sol[start]==-1){**  **cout<<"\nNo solution ";**  **return 0;**  **}**  **cout<<"\nCoin need to used to change money is: ";**  **while(start!=0){**  **int j=sol[start];**  **cout<<coin[j]<<" ";**  **start-=coin[j];**  **}**  **}** |

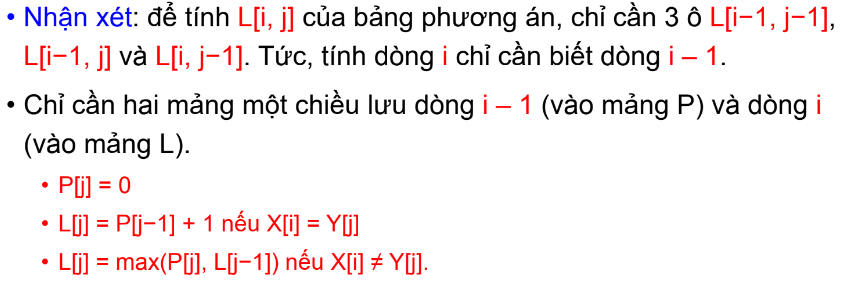
## **Bài 2. Xâu con chung dài nhất Cho hai xâu X và Y. Hãy tìm độ dài của xâu con chung dài nhất của X và Y, biết rằng xâu con gồm các ký tự không cần liên tiếp.**

*\*Ý tưởng*







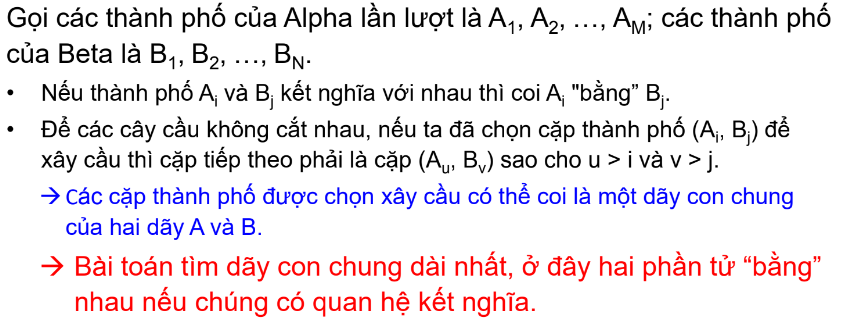


\*CODE

|  |
| --- |
| #include<bits/stdc++.h>  using namespace std;  int main(){  cout<<"String 1: ";  string a,b;  cin>>a;  cout<<"String 2: ";  cin>>b;  int n=a.length(),m=b.length();  int dp[n+1][m+1];  for(int i=0;i<=n;i++)  dp[i][0]=0;  for(int i=0;i<=m;i++)  dp[0][i]=0;  for(int i=1;i<=n;i++)  for(int j=1;j<=m;j++)  {  if(a[i-1]==b[j-1])  dp[i][j]=dp[i-1][j-1]+1;  else  dp[i][j]=max(dp[i-1][j],dp[i][j-1]);  }  for(int i=0;i<=n;i++){  for(int j=0;j<=m;j++)  cout<<dp[i][j]<<" ";  cout<<endl;  }  cout<<"Size of longgest common string is: "<<dp[n][m];  string res="";  while(dp[n][m]!=0){  if(a[n-1]==b[m-1]){  res=a[n-1]+res;  n--;  m--;  }  else{  if(dp[n-1][m]>dp[n][m-1])  n--;  else  m--;  }  }  cout<<"\nResult: "<<res;  } |

## **Bài 3: Hai nước Alpha và Beta nằm ở hai bên bờ sông Omega, Alpha nằm ở bờ bắc và có M thành phố được đánh số từ 1 đến M, Beta nằm ở bờ nam và có N thành phố được đánh số từ 1 đến N (theo vị trí từ đông sang tây). Mỗi thành phố của nước này thường có quan hệ kết nghĩa với một số thành phố của nước kia. Để tăng cường tình hữu nghị, hai nước muốn xây các cây cầu bắc qua sông, mỗi cây cầu sẽ là nhịp cầu nối 2 thành phố kết nghĩa. Với yêu cầu là các cây cầu không được cắt nhau và mỗi thành phố chỉ là đầu cầu cho nhiều nhất là một cây cầu, hãy chỉ ra cách bắc cầu được nhiều cầu nhất**

*\*Ý tưởng*



\*CODE

|  |
| --- |
| #include<bits/stdc++.h>  using namespace std;  int main(){  int n,m;  cout<<"Nhap n phan tu: ";  cin>>n;  cout<<"Nhap m phan tu: ";  cin>>m;  int a[n],b[m];  cout<<"Nhap a: ";  for(int i=0;i<n;i++){  cin>>a[i];  }  cout<<"Nhap b: ";  for(int i=0;i<m;i++){  cin>>b[i];  }  int dp[n+1][m+1];  for(int i=0;i<=n;i++){  dp[i][0]=0;  }  for(int i=0;i<=m;i++){  dp[0][i]=0;  }  cout<<endl;  for(int i=1;i<=n;i++){  for(int j=1;j<=m;j++){  if(a[i-1]==b[j-1])  dp[i][j]=dp[i-1][j-1]+1;  else  dp[i][j]=max(dp[i-1][j],dp[i][j-1]);  }  }  for(int i=1;i<=n;i++){  for(int j=1;j<=m;j++){  cout<<dp[i][j]<<" ";  }  cout<<endl;  }  cout<<"\nCau bac chung toi da la: "<<dp[n][m];  int u=max(n,m);  int res[u];  int dem=0;  while(dp[n][m]!=0){  if(a[n-1]==b[m-1]){  res[dem++]=a[n-1];  n--;  m--;  }  else{  if(dp[n-1][m]>=dp[n][m-1])  n--;  else  m--;  }  }  cout<<"\nCac cau xay la: ";  for(int i=0;i<dem;i++)  cout<<res[i]<<" ";  } |

