

ĐỀ THI CHÍNH THỨC

Môn thi: TIN HỌC
Buổi thi thứ hai: Chiều 11/10/2020

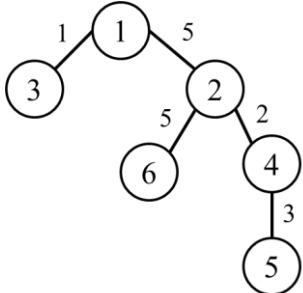
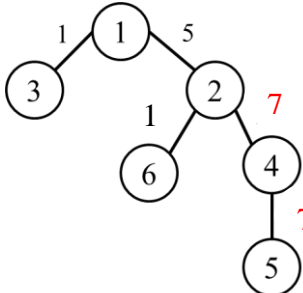
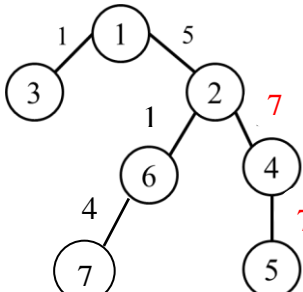
HƯỚNG DẪN CHẤM THI
(Bản hướng dẫn này gồm 10 trang)

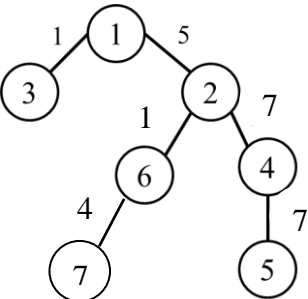
I. Hướng dẫn chung

- Mỗi bài thi của thí sinh sẽ được chấm bằng cách:
 - Chạy chương trình của thí sinh với bộ dữ liệu vào của hướng dẫn chấm; chương trình của thí sinh sẽ xuất ra các kết quả (gọi là kết quả của thí sinh) tương ứng các dữ liệu vào.
 - Đối chiếu sự trùng khớp giữa các kết quả của thí sinh với các kết quả của hướng dẫn chấm.
- Việc chạy chương trình của thí sinh và đối chiếu sự trùng khớp kết quả có thể thực hiện thủ công hoặc bằng phần mềm chấm thi. Ngoài ra, phần mềm chấm thi sẽ tính thời gian chạy chương trình nếu đề bài có yêu cầu.
- Trường hợp bài làm của thí sinh có lỗi (không biên dịch được), phần tổ chức dữ liệu và nhập dữ liệu được chấm tối đa 25% số điểm của bài.

II. Đáp án và thang điểm

BÀI	ĐÁP ÁN		ĐIỂM
Bài 4	Doanh thu <u>Cách 1</u> : Duyệt toàn bộ các dãy con liên tiếp của dãy đã cho (bằng hai vòng lặp lồng nhau), với mỗi dãy con, kiểm tra điều kiện “tổng $\geq S$ ”; nếu có, tính độ dài dãy con và cập nhật lại độ dài ngắn nhất. Độ phức tạp thuật toán $O(n^3)$. <u>Cách 2</u> : Cải tiến cách 1 bằng phương pháp chuẩn bị trước dữ liệu sao cho việc tính tổng chỉ tốn chi phí $O(1)$. Độ phức tạp thuật toán giảm còn $O(n^2)$. <u>Cách 3</u> : Cải tiến cách 2 bằng cách thay một vòng lặp bằng tìm kiếm nhị phân. Độ phức tạp thuật toán giảm còn $O(n \log n)$.		6,0
Test	DOANH THU.INP	DOANH THU.OUT	
1	6 15 0 0 4 0 5 6	4	Mỗi Test 0.5 điểm
2	6 50 20 0 -2 0 -5 6	-1	
3	10 90 9 9 9 9 9 9 9 9 9	10	
4	10 9 0 9 0 9 0 9 0 9 0 9	1	
5	File_input05	File_output05	
6	File_input06	File_output06	
7	File_input07	File_output07	

8	File_input08	File_output08	
9	File_input09	File_output09	
10	File_input10	File_output10	
11	File_input11	File_output11	
12	File_input12	File_output12	
Bài 5	Chạy Marathon <u>Cách 1:</u> Tìm đường đi ngắn nhất giữa các cặp đỉnh (Floyd). Tổ chức lưu giữ độ dài đường đi và số cạnh trên đường đi. Phân tích kết quả lưu giữ tìm lời giải. $O(n^3)$ <u>Cách 2:</u> Với mỗi đỉnh của đồ thị, duyệt đồ thị bằng BFS, biểu diễn đồ thị bằng ma trận kề. Độ phức tạp thuật toán $O(n^3)$. Có thể biểu diễn đồ thị bằng danh sách kề, độ phức tạp thuật toán còn $O(n^2)$. Có thể cải tiến thuật toán bằng các đánh giá nhánh cận theo các tiêu chí: - Chỉ duyệt đồ thị có trọng số không quá K. - Ở mỗi bước loang, không thăm đỉnh tiếp theo nếu thăm đỉnh ấy thì: + Chiều dài đường đi lớn hơn K; + Hoặc số cạnh trên đường đi <i>lớn hơn</i> số cạnh <i>tối ưu</i> đang có.		7,0
Test	MARATHON.INP	MARATHON.OUT	
1	6 11 1 2 5 1 3 1 2 4 2 2 6 5 4 5 3 	3 Giải thích: 3 – 1 – 2 - 6	Mỗi Test 0.5 điểm
2	6 6 1 2 5 1 3 1 2 4 7 2 6 1 4 5 7 	2 Giải thích: 3 – 1 – 2 hoặc 1 – 2 - 6	
3	7 5 1 2 5 1 3 1 2 4 7 2 6 1 4 5 7 6 7 4 	1 Giải thích: 1 – 2 <i>Giả sử khi duyệt từ đỉnh 1 đã tìm được đáp số tạm thời là 1. Sau đó duyệt từ đỉnh 2, ta thấy bước duyệt từ đỉnh 6 đến đỉnh 7 là không cần thiết.</i>	

4	7 26 1 2 5 1 3 1 2 4 7 2 6 1 4 5 7 6 7 4	-1 	<p><i>Giải thích:</i> Tổng các trọng số chỉ bằng 25, nhưng K đòi hỏi bằng 26, nên không tìm được lời giải.</p>
5	File_input05	File_output05	
6	File_input06	File_output06	
7	File_input07	File_output07	
8	File_input08	File_output08	
9	File_input09	File_output09	
10	File_input10	File_output10	
11	File_input11	File_output11	
12	File_input12	File_output12	
13	File_input13	File_output13	
14	File_input14	File_output14	
Bài 6	Cổ phiếu <u>Cách 1:</u> Duyệt toàn bộ bằng 2 vòng lặp. Độ phức tạp $O(n^2)$ <u>Cách 2:</u> Sử dụng cấu trúc dữ liệu Interval tree hoặc Binary Index tree. Độ phức tạp $O(n \log n)$		7,0
Test	COPHIEU.INP	COPHIEU.OUT	
1	5 10 10 10 10 10 5 0 3 12 1 1 3 0 3 15 1 3 5 1 1 3	12 15 15 CT: 0 3 12 → 10 10 12 10 10 CT: 1 1 3 → 12 CT: 0 3 15 → 10 10 15 10 10 CT: 1 3 5 → 15 CT: 1 1 3 → 15	Mỗi Test 0,5 điểm
2	5 10 20 30 40 50 5 1 1 5 0 1 40 1 1 3 0 3 60 1 1 5	50 40 60 CT: 1 1 5 → 50 CT: 0 1 40 → 40 20 30 40 50 CT: 1 1 3 → 40 CT: 0 3 60 → 40 20 60 40 50 CT: 1 1 5 → 60	

3	10 10 20 30 40 50 20 40 80 10 0 8 1 5 6 1 3 5 1 6 9 1 1 5 0 1 40 1 1 3 0 10 100 1 10 10	50 50 80 50 40 100 CT: 1 5 6 → 50 CT: 1 3 5 → 50 CT: 1 6 9 → 80 CT: 1 1 5 → 50 CT: 0 1 40 → 40 20 30 40 50 20 40 80 10 0 CT: 1 1 3 → 40 CT: 0 10 100 → 40 20 30 40 50 20 40 80 10 100 CT: 1 10 10 → 100	
4	10 1 1 1 1 1 1 1 1 1 1 5 0 3 6 0 2 2 0 9 5 0 2 5 1 2 9	6 CT: 0 3 6 → 1 1 6 1 1 1 1 1 1 1 CT: 0 2 2 → 1 2 6 1 1 1 1 1 1 1 CT: 0 9 5 → 1 2 6 1 1 1 1 1 5 1 CT: 0 2 5 → 1 5 6 1 1 1 1 1 5 1 CT: 1 2 9 → 6	
5	File_input05	File_output05	
6	File_input06	File_output06	
7	File_input07	File_output07	
8	File_input08	File_output08	
9	File_input09	File_output09	
10	File_input10	File_output10	
11	File_input11	File_output11	
12	File_input12	File_output12	
13	File_input13	File_output13	
14	File_input14	File_output14	

HẾT

Code tham khảo

Bài 4. Doanh thu

```
uses math;
const
  maxN = 100000;
  oo = 1000000000;
var
  a, vitri: array[1..maxN] of longint;
  sum, sum_sort: array[0..maxN] of longint;
  n, s: longint;
  j_min, luui, ans: longint;
//-----
procedure nhap;
var i: longint;
begin
  readln(n, s);
  for i:=1 to n do read(a[i]); // O(n)
end;
//-----
procedure doicho(var a, b: longint);
var tam: longint;
begin
  tam := a; a := b; b := tam;
end;
//-----
Procedure QuickSort(L, H: longint);
Var i, j, chot: longint;
begin
  If L >= H then exit;
  chot := sum_sort[(L+H) div 2];
  i := L; j := H;
  repeat
    while sum_sort[i] < chot do inc(i);
    While sum_sort[j] > chot do dec(j);
    if i <= j then
      begin
        if i < j then begin doicho(sum_sort[i], sum_sort[j]); doicho(vitri[i], vitri[j]) end;
        inc(i); dec(j);
      end;
  until i > j;
  QuickSort(L, j);
  QuickSort(i, H);
end;
//-----
procedure chuanbi; // O(nlogn)
var i: longint;
begin
  fillchar(sum, sizeof(sum), 0);
```

```

for i:=1 to n do // O(n)
begin
    sum[i] := sum[i-1]+a[i];
    vitri[i] := i;
end;
sum_sort := sum;
QuickSort(1,n); // O(nlogn)
end;
//-----
procedure chuanbi2;
var i: longint;
begin
    fillchar(sum,sizeof(sum),0);
    for i:=1 to n do
        begin
            sum[i] := sum[i-1] + a[i];
        end;
end;
//-----
procedure tim(x,a,b:longint); // O(logn)
var L,H,Mid: longint;
begin
    if a>b then exit;
    L := a; H := b;
    Mid := (L+H) div 2;
    if sum_sort[Mid] >= x then
        begin
            if (vitri[Mid]<j_min) and (vitri[Mid] >= luui) then j_min := vitri[Mid];
            if (Mid<H) then tim(x,Mid+1,H);
            if (L<Mid) and (sum_sort[Mid-1] >= x) then tim(x,L,Mid-1); // danh gia nhanh can
        end
    else
        begin
            if (Mid<H) then tim(x,Mid+1,H);
        end;
end;
//-----
procedure xuli;
var i: longint;
begin
    chuanbi; // chuan bi truoc du lieu
    ans := oo;
    for i:=1 to n do // O(n)
        begin
            j_min := n+1;
            luui := i;
            tim(sum[i-1]+s,i,n); // O(logn)
            if (j_min >= i) and (j_min <= n) then

```

```

        begin
            if (j_min-i+1) < ans then ans := (j_min-i+1);
        end;
    end;
    if (ans=oo) then writeln(-1) else writeln(ans);
end;
//-----
procedure xuli2;
var i,j: longint; tong: int64;
begin
    ans := oo;
    chuanbi2;
    for i:=1 to n do
        for j:=i to n do
            begin
                tong := sum[j] - sum[i-1];
                if tong >= s then
                    begin
                        if (j-i+1) < ans then ans := j - i + 1;
                    end;
                end;
            end;
        if (ans=oo) then writeln(-1) else writeln(ans);
    end;
//-----
BEGIN {Main}
    assign(input,'doanhthu.inp'); reset(input);
    assign(output,'doanhthu.out'); rewrite(output);
    nhap;
    //xuli2;
    xuli;
    close(input); close(output);
END.

```

Bài 5. Chạy Marathon

```

const
    maxN = 200000;
    oo = 2000000000;
type
    link = ^node;
    node = record
        dinh: longint;
        L: longint;
        next: link;
    end;
var
    Ke: array[1..maxN] of link;
    Queue: array[1..3*maxN] of longint;
    Free: array[1..maxN] of Boolean;
    D,KM: array[1..MaxN] of longint;

```

```

    first,last,n, K:longint;
    SoCanh_Min,ans: longint;

//-----
procedure add(u,v,L:longint);
var p,q: link;
begin
    new(p);
    p^.dinh := v; p^.L := L; p^.next := ke[u]; ke[u] := p;
    new(q);
    q^.dinh := u; q^.L := L; q^.next := ke[v]; ke[v] := q;
end;
//-----
procedure Push(v:longint);
begin
    inc(last);
    Queue[last]:=v;
end;
//-----
function pop:longint;
begin
    pop:=Queue[first];
    inc(first);
end;
//-----
procedure BFS(s:longint);
var u,v:longint; p: link;
begin
    first:=1; last:=0; {khởi tạo hàng đợi rỗng}
    Fillchar(Free,sizeof(Free),True);
    Fillchar(D,sizeof(D),0);
    Fillchar(KM,sizeof(KM),0);
    Free[s] := False;

    push(s);
    while first<=last do
        begin
            u:=pop;
            p := ke[u];
            while (p<>nil) do
                begin
                    v := p^.dinh;
                    if Free[v] then
                        begin
                            Free[v] := false;
                            D[v] := D[u]+1;
                            KM[v] := KM[u] + p^.L;
                            if KM[v]= K then

```



```

        begin socanh_min := D[v]; exit end;
        if KM[v] < K then push(v); // nhanh can
        end;
        p := p^.next;
    end;

    end;
end;
//-----
procedure nhap;
var i,u,v,L:longint;
begin
    readln(n,K);
    for i:=1 to n-1 do
        begin
            readln(u,v,L);
            add(u,v,L);
        end;
    end;
end;

//-----
procedure xuli;
var s: longint;
begin
    ans := oo;
    socanh_min := oo;
    for s:=1 to n do
        begin
            BFS(s);
            if socanh_min < ans then ans := socanh_min;
        end;
    end;
    if ans = oo then write(-1) else write(ans);
end;
//-----
BEGIN
    assign(Input,'marathon.inp'); reset(Input);
    assign(Output,'marathon.out'); rewrite(Output);
    nhap;
    xuli;
    Close(Input);
    Close(Output);
END.

```

Bài 6. Cổ phiếu

```

uses math;
const
    maxN = 1000000; maxM = 100000;
    oo = 2000000000;
var

```

```

a,pos: array[1..maxN] of longint;
tree: array[1..100*maxN] of longint;
n,p,u,v,max_uv: longint;

//-----
Procedure dungcay(k,be,en: longint);
Var mid : longint;
Begin
  If (be = en) then begin tree[k] := a[be]; pos[be] := k end
  Else
    begin
      mid := (be+en) div 2 ;
      dungcay(2*k,be,mid);
      dungcay(2*k+1,mid+1,en);
      tree[k] := max(tree[2*k],tree[2*k+1]);
    end;
End;
//-----
procedure truyvan(k,be,en,L,R: longint);
Var mid: longint;
Begin
  if (en<L) or (R<be) then exit;
  if (L<=be) and (en<=R) then max_uv := max(max_uv,tree[k])
  else
    begin
      mid := (be+en) div 2;
      truyvan(2*k,be,mid,L,R);
      truyvan(2*k+1,mid+1,en,L,R);
    end;
End;
//-----
procedure capnhathat(u,v: longint);
var k: longint;
begin
  k := pos[u];
  a[u] := v;
  tree[k] := v;
  k := k div 2;
  repeat
    if (k=0) or (tree[k] >= max(tree[2*k],tree[2*k+1])) then break; // danh gia nhanh can
    tree[k] := max(tree[2*k],tree[2*k+1]);
    k := k div 2;
  until false;
end;
//-----
procedure nhap;
var i,loai,u,v: longint;
begin

```

```

readln(n);
for i:=1 to n do read(a[i]);
dungcay(1,1,n);
readln;
readln(p);
for i:=1 to p do
  begin
    readln(loai,u,v);
    if (loai = 0) then capnhat(u,v) else
      begin
        max_uv := -oo;
        truyvan(1,1,n,u,v);
        writeln(max_uv);
      end;
  end;
end;
//-----
BEGIN
  assign(input,'cophieu.inp'); reset(input);
  assign(output,'cophieu.out'); rewrite(output);
  nhap;
  close(input); close(output);
END.

```