SỞ GIÁO DỤC VÀ ĐÀO TẠO
HẢI DƯƠNG
-------- 🕮 -------

ĐÁP ÁN CHÍNH THỨC

KỲ THI CHỌN HỌC SINH GIỎI CẤP TỈNH LỚP 12 THPT
NĂM HỌC 2017-2018
MÔN: TIN HỌC
Thời gian làm bài: **180** phút (không kể thời gian giao đề)
(Đáp án gồm **07** trang)

Tổng quan về các bài thi trong đề:

TT	Tên bài	Tên file chương trình	Dữ liệu vào	Dữ liệu ra	Thời gian chạy 1 test	Điểm
1	Trả tiền nước	BAI1.*	bàn phím	màn hình	1s	2,5
2	Đếm sách	BAI2.*	bàn phím	màn hình	1s	2,5
3	Bạn bè	BAI3.*	bàn phím	màn hình	1s	2,0
4	Dãy dài nhất	BAI4.*	BAI4.INP	BAI4.OUT	1s	1,5
5	Chia phần	BAI5.*	BAI5.INP	BAI5.OUT	1s	1,5

Yêu cầu các thí sinh đọc kỹ phần hướng dẫn dưới đây:

- *Dấu (*) trong tên file chương trình được thay thế bằng PAS hoặc CPP tuỳ theo thí sinh viết chương trình bằng ngôn ngữ Pascal hoặc C++*
- *Chương trình chỉ in kết quả theo yêu cầu của đề bài, không in bất kỳ thông tin nào khác*
- *Chương trình sử dụng lệnh in (**write, writeln** đối với Pascal; **printf(…), cout** đối với C++) để in kết quả*
- *Đối với các bài tập đọc và in dữ liệu từ file văn bản, tên các file này phải đặt đúng theo yêu cầu đề bài, không có đường dẫn phía trước.*

Viết chương trình giải các bài toán sau:

Bài 1: Trả tiền nước

Công ty TNHH MTV kinh doanh nước sạch trên địa bàn một tỉnh quy định giá bán nước sạch sinh hoạt cho các hộ dân cư trong địa bàn tỉnh như sau:

Lượng nước sạch sử dụng (hộ/tháng)	Giá bán (đ/m^3)
Từ m^3 thứ 1 đến m^3 thứ 5	6500
Từ m^3 thứ 6 đến m^3 thứ 15	7800
Từ m^3 thứ 16 đến m^3 thứ 25	9200
Từ m^3 thứ 26 trở lên	10300

(Giá bán trên chưa bao gồm thuế VAT và phí nước thải)

Tính số tiền phải trả cho công ty nước sạch của một hộ gia đình trong một tháng, biết rằng thuế VAT và phí nước thải là 12%.

Dữ liệu: Nhập từ bàn phím số nguyên dương $(0 < \quad \leq 1000)$ là số m^3 nước sạch mà một hộ gia đình dùng trong một tháng.

Kết quả: In ra màn hình ba giá trị tương ứng trên ba dòng, mỗi số gồm hai chữ số thập phân.

- *Dòng 1*: Số tiền tương ứng với giá bán nước của công ty.
- *Dòng 2*: Số tiền tương ứng với thuế VAT và phí nước thải.
- *Dòng 3:*Tổng số tiền nước mà hộ gia đình đó phải trả trong tháng đó.

Ví dụ:

Dữ liệu vào	Dữ liệu ra
5	32500.00 3900.00 36400.00

Ghi chú: Bài được chấm qua 10 test, mỗi test đúng được 0,25 điểm.

Thuật toán:
Đây là bài tập kiểm tra kiến thức cơ bản về lập trình (sử dụng cấu trúc rẽ nhánh). Dưới đây là chương trình tham khảo:

```
var
   N, G, VAT: double;

BEGIN
   read(N);
   if N<=5 then G:=N*6500 else
   if N<=15 then G:=5*6500+(N-5)*7800 else
   if N<=25 then G:=5*6500+10*7800+(N-15)*9200
   else G:=5*6500+10*7800+10*9200+(N-25)*10300;
   VAT:=G*12/100;
   writeln(G:0:2);
   writeln(VAT:0:2);
   writeln(G+VAT:0:2);
END.
```

Bài 2. Đếm sách

Trong một cửa hiệu bán sách. Để dễ quản lý các loại sách có trong hiệu sách, người bán hàng đã gán tương ứng mỗi loại sách với một số nguyên dương, hai loại sách khác nhau có số được gán là hai số nguyên khác nhau. Em hãy viết chương trình giúp chủ cửa hiệu tìm loại sách còn nhiều nhất và số lượng còn là bao nhiêu.

Dữ liệu: Nhập từ bàn phím số (≤ 100) là số lượng sách còn lại trong cửa hiệu, tiếp theo là số nguyên mô tả loại sách của quyển sách này, hai số nguyên liên tiếp cách nhau một dấu trống. Giá trị các số nguyên không vượt quá 10 .

Kết quả: In ra màn hình trên một dòng số xuất hiện nhiều nhất và số lần xuất hiện của nó, hai giá trị này in cách nhau một dấu trống. Nếu như có nhiều số có số lần xuất hiện nhiều nhất thì in số có giá trị bé nhất.

Ví dụ:

Dữ liệu nhập	Dữ liệu xuất
11 1 2 2 3 2 4 5 2 6 7 6	2 4

Ghi chú: Bài được chấm qua 10 test, mỗi test đúng được 0,25 điểm trong đó

- 6 test có giá trị các mã số trong phạm vị từ 1 đến 1000
- 4 test có giá trị các mã số trong phạm vi từ 1 đến 10^9.

Thuật toán:

Thuật toán chính của bài toán là với mỗi giá trị mã, đếm xem có bao nhiêu loại sách có mã bằng giá trị này, từ đó cập nhật mã có số lần xuất hiện nhiều nhất. Dưới đây là chương trình tham khảo:

```
var N: longint;
    a: array[0..101] of longint;
    dapso, soluong, ten, nhat: longint;
BEGIN
   read(N);
   for i:=1 to N do read(a[i]);
   dapso:=0; nhat=0;
   for i:=1 to n do
   begin
      soluong:=0;
      for j:=1 to n do if a[i]=a[j] then inc(soluong);
      if (soluong>dapso) or ((soluong=dapso) and (a[i]<nhat)) then
      begin
         dapso:=soluong;
         nhat:=a[i];
      end;
   end;
   writeln(nhat,' ',dapso);
END.
```

Chú ý: trong các test có 6 test ứng với giá trị mã trong khoảng [1,1000] thể hiện rằng nếu học sinh sử dụng số nguyên kiểu 2 byte (integer trong Pascal, sortint trong C++) vẫn qua được các test này

Bài 3. Bạn bè

Hai từ gọi là bạn bè nếu chúng được tạo nên bởi cùng một tập hợp kí tự giống nhau: Ví dụ **S1='aabbbccccb'** và **S2='aabccccaaaaa'** là bạn bè vì nó cùng được tạo bởi tập ký tự **{'a','b','c'}**. Cho ba cặp hai từ; với mỗi cặp in 'YES' nếu hai từ trong cặp là bạn bè và in 'NO' nếu chúng không phải là bạn bè.

Dữ liệu: Nhập từ bàn phím 6 xâu ký tự (mô tả 6 từ) lần lượt là S1, S2, S3, S4, S5, S6; mỗi xâu trên một dòng chỉ gồm chữ cái tiếng Anh in thường có độ dài không vượt quá 1000

Kết quả: In ra ba dòng:

- Dòng 1: In 'YES' nếu S1 và S2 là bạn bè, ngược lại in 'NO'
- Dòng 2: In 'YES' nếu S3 và S4 là bạn bè, ngược lại in 'NO'
- Dòng 3: In 'YES' nếu S5 và S6 là bạn bè, ngược lại in 'NO'

Ví dụ:

Dữ liệu vào	Dữ liệu ra
aabbbccccb	YES
aabccccaaaaa	NO
aabbbccccbcc	YES
aadddccccaaa	
xyzabc	
aaaaxxyxxzcccb	

Ghi chú: Bài được chấm qua 8 test, mỗi test đúng được 0,25 điểm:

- 6 test có độ dài của mỗi xâu ký tự không vượt quá 255
- 2 test có độ dài của mỗi xâu ký tự không vượt quá 1000

Thuật toán:

Ta biểu diễn loại của một dãy ký tự bằng tập hợp các ký tự xuất hiện trong dãy. Để làm điều này có thể mô tả tập hợp ký tự của một dãy bằng một mảng:
c: array[‘a’..’z’] of integer;
Trong đó c[i]=1/0 tùy theo ký tự i có xuất hiện trong xâu ký tự hay không. Việc hai xâu ký tự là bạn bè đơn giản chỉ là việc so sánh hai mảng có bằng nhau hay không?
Nếu coi độ dài các xâu ký tự không quá 255 ta có thể sử dụng kiểu **string** trong Pascal. Dưới đây là chương trình minh họa:

```
var s1, s2: string;
    a, b: array[‘a’..’z’] of longint;

function ok: boolean;
var i: char;
begin
   for i:=’a’ to ‘z’ do if a[i]<>b[i] then exit(false);
   exit(true);
end;

var i, t: longint;
    c: char;

BEGIN
   for t:=1 to 3 do
   begin
      readln(s1);
      readln(s2);
      for c:=’a’ to ‘z’ do
      begin a[c]:=0; b[c]:=0; end;
      for i:=1 to length(s1) do inc(a[s1[i]]);
      for i:=1 to length(s2) do inc(b[s2[i]]);
      if ok then writeln('YES') else writeln('NO');
   end;
END.
```

Chương trình trên qua được 6 test. Để có thể qua được các test còn lại ta cải tiến: thay vì dùng kiểu xâu ký tự (string) ta thực hiện việc xử lý ngay ký tự khi đọc vào (không lưu thành xâu). Dưới đây là chương trình minh họa:

```
var ch: char;
    a, b: array[‘a’..’z’] of longint;

function ok: boolean;
var i: char;
begin
   for i:=’a’ to ‘z’ do if a[i]<>b[i] then exit(false);
   exit(true);
end;

var i, t: longint;
    c: char;

BEGIN
   for t:=1 to 3 do
  begin
      for c:=’a’ to ‘z’ do
      begin a[c]:=0; b[c]:=0; end;
while not seekeoln do
      begin
         read(c);
         inc(a[c]);
      end;
      readln;
while not seekeoln do
      begin
         read(c);
         inc(b[c]);
      end;
      readln;
      if ok then writeln('YES') else writeln('NO');
   end;
END.
```

Nếu sử dụng Free Pascal học sinh có thể sử dụng kiểu **ansistring** thay cho **string** và vẫn được 100% số điểm của bài

Bài 4. Dãy dài nhất

Cho dãy số nguyên dương $= (\quad , \quad , \ldots , \quad)$ và số nguyên dương . Hãy tìm dãy con dài nhất (là dãy có nhiều số nhất) gồm các số liên tiếp của A mà tổng tất cả các số của dãy con này chia hết cho .

Dữ liệu: Nhập từ file văn bản BAI4.INP

- Dòng đầu tiên ghi hai số nguyên dương , (≤ 10 , ≤ 10) ghi cách nhau một dấu trống
- Dòng thứ hai ghi số nguyên dương , , ..., mô tả dãy , hai số nguyên liên tiếp ghi cách nhau một dấu trống. Giá trị các số nguyên không vượt quá 10^9

Kết quả: Ghi ra file văn bản BAI4.OUT độ dài của dãy con dài nhất tìm được

Ví dụ:

BAI4.INP	BAI4.OUT

6 3 3 2 4 6 3 7	5

Ghi chú: Kết quả được chấm qua 6 test, mỗi test đúng được 0,25 điểm, trong đó:

- 2 test có N≤500
- 2 test có N≤5000
- 2 test có N≤10^5

Thuật toán:

Với 2 test có N≤500 ta có thuật toán đơn giản: Thử hết tất cả các dãy con a_i, a_{i+1},...,a_j . Với mỗi dãy con tìm được tính tổng các số. Nếu tổng này chia hết cho K thì so sánh độ dài của dãy với đáp số (kết quả tốt nhất) đang lưu trữ. Dưới đây là chương trình minh họa:

```
var
   n: longint;
   a: array[0..100001] of longint;
function tong(i,j: longint): longint;
var t, u: longint;
begin
   t:=0;
   for u:=i to j do t:=t+a[u];
   exit(t);
end;
var i, j, S: longint;
BEGIN
   assign(input,'BAI4.INP'); aeset(input);
   assign(output,'BAI4.OUT'); rewrite(output);
   read(n);
   for i:=1 to n do read(a[i]);
   dapso:=0;
   for i:=1 to n do
   for j:=i to n do
   begin
      S:=tong(i,j);
      if (S mod K=0) and (dapso<j-i+1) then dapso:=j-i+1;
   end;
   writeln(dapso);
END.
```

Trên đây chỉ là chương trình minh họa, trong thực tế khi code cần lưu ý rằng giá trị trả về của hàm tong thường vượt quá 2.10^9 nên cần sử dụng kiểu số nguyên 64 bit (int64 trong Free Pascal) hoặc kiểu số thực.

Độ phức tạp của thuật toán trên là $O(n^3)$ nên không thể qua được các test có N≤5000. Để qua được các test này cần chú ý lập các mảng "tổng tiền tố":

s[0]=0; s[i]=s[i-1]+a[i] với i=1,2,...,n

Nếu như có mảng này thì tổng các phần tử trong đoạn a_i, a_{i+1}, ..., a_j được tính bởi s[j]-s[i-1] và ta có thuật toán với độ phức tạp $O(n^2)$ như chương trình minh họa dưới đây:

```
var
   n: longint;
   a, s: array[0..100001] of longint;
```

```
var i, j, S: longint;
BEGIN
   assign(input,'BAI4.INP'); aeset(input);
   assign(output,'BAI4.OUT'); rewrite(output);
   read(n);
   for i:=1 to n do read(a[i]);
   s[0]:=0;
   for i:=1 to n do s[i]:=s[i-1]+a[i];
   dapso:=0;
   for i:=1 to n do
   for j:=i to n do
   begin
      S:=s[j]-s[i-1];
      if (S mod K=0) and (dapso<j-i+1) then dapso:=j-i+1;
   end;
   writeln(dapso);
END.
```

Với thuật toán $O(n^2)$ ta giải quyết được với $N \le 5000$. Tuy nhiên khi $N \le 10^5$ ta cần phải có một cách tiếp cận khác: Lưu ý rằng trong cả hai thuật toán trên ta chưa sử dụng điều kiện $K \le 10^5$ của đề bài. Ngoài ra chú ý rằng:

S mod K = (S[j]-S[i-1]) mod K

Nên nếu S mod K=0 thì S[j] và S[i-1] phải cùng đồng dư khi chia cho K. Do vậy bài toán qui về là với mỗi j cần tìm vị trí đầu tiên đã có s[j] mod K. Do $K \le 10^5$ nên ta có thể sử dụng một mảng nho[0..100000] of longint; để nhớ xem mỗi số dư xuất hiện hay chưa. Chương trình dưới đây minh họa điều này:

```
var
   n: longint;
   a, s, nho: array[0..100001] of longint;
var i, j, S: longint;
BEGIN
   assign(input,'BAI4.INP'); aeset(input);
   assign(output,'BAI4.OUT'); rewrite(output);
   read(n);
   for i:=1 to n do read(a[i]);
   s[0]:=0;
   for i:=1 to n do s[i]:=s[i-1]+a[i];
   dapso:=0;
   for i:=0 to K do nho[i]=-1;
   nho[0]:=0;
   for i:=1 to n do
   begin
      u:=s[i] mod K;
      if (nho[u]<>-1) and (dapso<i-u+1) then dapso:=i-u+1;
if nho[u]=-1 then nho[u]:=i;
   end;
   writeln(dapso);
END.
```

Thuật toán có độ phức tạp O(n) và qua được 100% số test.

Bài 5. Chia phần

Cho dãy số nguyên $= (\ , \ , \ldots , \ )$; Hãy đếm số cách chia dãy trên thành 4 dãy con gồm các số liên tiếp của sao cho tổng các số trong mỗi dãy con đều bằng nhau. Chính xác hơn, mỗi cách chia được mô tả bằng bộ 3 chỉ số $(\ , \ , \ )$: $1 \leq \ < \ < \ < \ $. Trong đó $(\ , \ , \ldots , \ )$ là dãy 1; $(\ , \ , \ldots , \ )$ là dãy 2; $(\ , \ , \ldots , \ )$ là dãy 3 và $(\ , \ , \ldots , \ )$ là dãy 4. Hai cách chia khác nhau ứng với hai bộ 3 chỉ số $(\ , \ , \ )$ khác nhau.

Dữ liệu: Vào từ file văn bản BAI5.INP

- Dòng đầu tiên ghi số nguyên dương $(\ \leq 10 \ )$
- Dòng thứ hai ghi N số nguyên $\ , \ , \ldots , \ (| \ | \leq 10 \ ; \ = 1,2, \ldots , \ )$; hai số liên tiếp cách nhau bằng một dấu trống

Kết quả: Ghi ra file văn bản BAI5.OUT một số nguyên là số lượng cách chia tìm được

Ví dụ:

BAI5.INP	BAI5.OUT
8 1 1 1 1 1 1 1 1	1

Chú ý: Kết quả được chấm qua 6 test, mỗi test đúng được 0,25 điểm, trong đó:

- 2 test có ≤ 50
- 2 test có ≤ 500
- 1 test có ≤ 5000
- 1 test có ≤ 10

Thuật toán:
Phương án đơn giản nhất là thử tất cả các bộ (i,j,k) với 1≤i<j<k≤n. Với mỗi bộ trên tính các tổng: S1=a[1]+a[2]+…+a[i]; S2=a[i+1]+…+s[j]; S3=a[j+1]+…+a[k]; S4:=a[k+1]+…+a[n]. Nếu như bốn giá trị bằng nhau thì ta có một phương án chia. Chương trình dưới đây minh họa điều trên:

```
var
   n: longint;
   a: array[0..1000000] of longint;
   dapso: longint;

function tong(i,j: longint): longint;
var t, u: longint;
begin
   t:=0;
for u:=i to j do t:=t+a[u];
   exit(t);
end;

var i, j, k, S1, S2, S3, S4: longint;
BEGIN
   assign(input,'BAI5.INP'); reset(input);
   assign(output,'BAI5.OUT'); rewrite(output);
```

```
   read(n);
   for i:=1 to n do read(a[i]);
   dapso:=0;
   for i:=1 to n do
   for j:=i+1 to n do
   for k:=j+1 to n do
   begin
      S1:=tong(1,i); S2:=tong(i+1,j);
      S3:=tong(j+1,k); S4:=tong(k+1,n);
      if (S1=S2) and (S2=S3) and (S3=S4) then inc(dapso);
   end;
   writeln(dapso);
END.
```

Thuật toán trên có độ phức tạp $O(n^4)$ và ta qua được 2 test đầu. Để qua được 2 test tiếp theo cần lập mảng tổng tiền tố:
S[0]=0; S[i]=S[i-1]+a[i];
Và không cần phải hàm tính tổng. Chương trình dưới đây minh họa điều này:

```
var
   n: longint;
   a, S: array[0..1000000] of longint;
   dapso: longint;
var i, j, k, S1, S2, S3, S4: longint;
BEGIN
   assign(input,'BAI5.INP'); reset(input);
   assign(output,'BAI5.OUT'); rewrite(output);
   read(n);
   for i:=1 to n do read(a[i]);
   S[0]:=0;
   for i:=1 to n do S[i]:=S[i-1]+a[i];
   dapso:=0;
   for i:=1 to n do
   for j:=i+1 to n do
   for k:=j+1 to n do
   begin
      S1:=S[i]; S2:=S[j]-S[i-1];
      S3:=S[k]-S[j-1]; S4:=S[n]-S[k-1];
      if (S1=S2) and (S2=S3) and (S3=S4) then inc(dapso);
   end;
   writeln(dapso);
END.
```

Chương trình trên có độ phức tạp $O(n^3)$ và qua được các test có $N \leq 500$.
Để qua được các test có $N \leq 5000$ ta chú ý rằng S[n] phải chia hết cho 4 và các bộ (i,j,k) thỏa mãn chia được thành 4 phần bằng nhau phải có
S[i]=S[n] div 4; S[j]=2*(S[n] div 4); S[k]=3*(S[n] div 4)
Điều này dẫn đến kết luận rằng với mỗi giá trị j thỏa mãn S[j]=2*(S[n] div 4) chỉ cần đếm xem phía trước nó có bao nhiêu giá trị bằng S[n] div 4 và phía sau nó có bao nhiêu giá trị bằng 3*(S[n] div 4). Chương trình dưới đây minh họa ý tưởng này:

```
var
   n: longint;
   a, S: array[0..1000000] of longint;
   dapso: longint;
var i, j, k, p, q: longint;
BEGIN
   assign(input,'BAI5.INP'); reset(input);
   assign(output,'BAI5.OUT'); rewrite(output);
   read(n);
   for i:=1 to n do read(a[i]);
```

```
   S[0]:=0;
   for i:=1 to n do S[i]:=S[i-1]+a[i];
   dapso:=0;
   if s[n] mod 4=0 then
   begin
     for j:=2 to n-2 do if S[j]=2*(S[n] div 4) then
     begin
        p:=0;
        for i:=1 to j-1 do if S[i]=S[n] div 4 then inc(p);
        q:=0;
         for k:=j+1 to n-1 do if S[k]=3*(S[n] div 4) inc(q);
         dapso:=dapso+p*q;
      end;
   end;
   writeln(dapso);
END.
```

Độ phức tạp thuật toán trên là $O(n^2)$ và ta qua được các test có $N \leq 5000$.
Để qua được các test có $N \leq 10^6$ ta cần phải có thuật toán O(n). Xét thuật toán $O(n^2)$ ở trên. Nhận xét rằng p và q với mỗi j có thể được chuẩn bị từ trước và do vậy mỗi lần lặp ta chỉ cần trong O(1) là tính được hai giá trị này. Chương trình dưới minh họa tư tưởng trên:

```
var
   n: longint;
   a, S, p, q: array[0..1000000] of longint;
   dapso: longint;
var i, j, k: longint;
BEGIN
   assign(input,'BAI5.INP'); reset(input);
   assign(output,'BAI5.OUT'); rewrite(output);
   read(n);
   for i:=1 to n do read(a[i]);
   S[0]:=0;
   for i:=1 to n do S[i]:=S[i-1]+a[i];
   dapso:=0;
   if s[n] mod 4=0 then
   begin
     p[0]:=0;
     for i:=2 to n do
     begin
        p[i]=p[i-1];
        if s[i] = s[n] div 4 then inc(p[i]);
     end;
     q[n]:=0;
     for k:=n-1 downto 1 do
     begin
        q[k]:=q[k+1];
        if s[k]=3*(s[n] div 4) then inc(q[k]);
     end;
     for j:=2 to n-2 do if S[j]=2*(S[n] div 4) then
     dapso:=dapso+p[j-1]*q[j+1];
   end;
   writeln(dapso);
END.
```

Độ phức tạp thuật toán là O(n) và qua được 100% số test của bài

.....**The end**.....