# Завдання IІ етапу Всеукраїнської учнівської олімпіади

# з інформатики 2016-2017 н.р.

**Задача A. Олімпіада (100 балів)**

**Ім’я вхідного файлу: іnput.txt**

**Ім’я вхідного файлу: output.txt**

**Ліміт часу: 1с.**

З метою якісної підготовки до олімпіади з інформатики учень Степан виділив певний час кожного дня на підготовку. Допоможіть вчителю інформатики визначити загальний час який учень готовився до олімпіади.

**Вхідні дані**

У першому рядку вхідного файлу записано N кількість днів підготовки, в наступних двох рядках початковий та кінцевий час в форматі Г Х С (без початкових нулів). При цьому він задовольняє обмеженням: Г - від 0 до 23, Х і С - від 0 до 60.

**Вихідні дані**

У вихідний файл виведіть в форматі Д Г Х С час, який Степан витратив на підготовку до олімпіади (де Д – кількість днів).

**Приклади**

|  |  |
| --- | --- |
| іnput.txt | output.txt |
| 113 10 015 30 10 | 0 2 20 10 |
| 200 0 012 0 0 | 10 0 0 0 |

|  |
| --- |
| #include <fstream>using namespace std;ifstream cin("input.txt");ofstream cout("output.txt");int main(){ long long n,g1,h1,s1, g2,h2,s2, d,g,h,s; cin>> n>>g1>>h1>>s1>>g2>>h2>>s2; long long t =n\*(g2\*3600+h2\*60+s2-(g1\*3600+h1\*60+s1)); d=t/(24\*3600); g=(t-d\*24\*3600)/3600; h=(t-d\*24\*3600-g\*3600)/60; s=(t-d\*24\*3600-g\*3600-h\*60); cout <<d<<" "<<g<<" "<<h<<" "<<s<< endl; return 0;} |

**Задача B. Скарб (100 балів)**

**Ім’я вхідного файлу: іnput.txt**

**Ім’я вхідного файлу: output.txt**

**Ліміт часу: 1с.**

Знайти закопаний піратами скарб просто: все, що для цього потрібно - це карта. Як відомо, пірати зазвичай малюють карти від руки і описують алгоритм знаходження скарбу так: «Встаньте близько самотньою пальми. Пройдіть тридцять кроків у бік лісу, потім сімнадцять кроків у бік озера, ..., нарешті десять кроків у бік великого каменю. Скарб знаходиться під ним ». Велика частина таких вказівок просто зводиться до проходження якоїсь кількості кроків в одному з восьми напрямків (1 - північ, 2 - північний схід, 3 - схід, 4 - південний схід, 5 - південь, 6 - південний захід, 7 - захід, 8 -. північний захід) (див рис). Довжина кроку в будь-якому напрямку дорівнює 1.

Подорож по такому шляху зазвичай є прекрасним способом подивитися околиці, однак в наш час постійного поспіху ні у кого немає часу на це. Тому шукачі скарбів хочуть йти безпосередньо в точку, де заритий скарб. Наприклад, замість того, щоб проходити три кроки на північ, один крок на схід, один крок на північ, три кроки на схід, два кроки на південь і один крок на захід, можна пройти безпосередньо, використавши близько 3.6 кроку (див. Рис)



Вам необхідно написати програму, яка за вказівками піратів визначає точку, де заритий скарб.

**Вхідні дані**

Перший рядок вхідного файлу містить число N - число вказівок (1≤N≤40). N Наступні рядків містять самі вказівки - номер напрямку (ціле число від 1 до 8) і кількість кроків (ціле число від 1 до 1000). Числа розділені пробілами.

Вихідні дані

У вихідний файл виведіть координати X і Y точки (два дійсних числа, розділені пробілом), де заритий скарб, вважаючи, що вісь Ox спрямована на схід, а вісь Oy - на північ. На початку скарбошукач повинен стояти на початку координат. Координати необхідно вивести з похибкою не більше 103.

**Приклад**

|  |  |
| --- | --- |
| іnput.txt | output.txt |
| 61 33 11 13 35 27 1 | 3.000 2.000 |
| 18 10 | -7.071 7.071 |

|  |  |
| --- | --- |
| #include <fstream>#include <math.h>using namespace std;ifstream cin("input.txt");ofstream cout("output.txt");int main(){ const double pi = atan(1.0)\*4; double X, Y,L; int N,D; cin>>N; X=0; Y=0; for(int i=1;i<=N;i++) { cin>>D>>L; X=X+sin(pi/4\*(D-1))\*L; Y=Y+cos(pi/4\*(D-1))\*L; }if (fabs(X)<0.001) X=0;if (fabs(Y)<0.001) Y=0; cout.precision(3); cout<<fixed<<X<<" "<<Y<<endl; return 0;} | {$N+,E-}Var I,N:LongInt; D,L:LongInt; X,Y:Extended;Begin Randomize; Assign(Input,'c.in'); ReSet(Input); Read(N); X:=0; Y:=0; For I:=1 To N Do Begin Read(D,L); X:=X+Sin(Pi/4\*(D-1))\*L; Y:=Y+Cos(Pi/4\*(D-1))\*L; End; Close(Input); Assign(Output,'c.out'); ReWrite(Output); WriteLn(X:0:3,' ',Y:0:3); Close(Output);End. |

**Задача С. Дошка (100 балів)**

**Ім’я вхідного файлу: іnput.txt**

**Ім’я вхідного файлу: output.txt**

**Ліміт часу: 1с.**

 У кожній клітинці шашкової дошки довільного розміру в довільному порядку знаходяться шашки одного з кольорів: чорна, біла, червона, зелена.

 Скласти програму, що підраховує кількість шашок кожного кольору і що виводить результат якщо дані шашки, знаходяться у файлі (Приклад 1, Приклад 2) в першому рядку міститься кількість рядків N, в наступних N рядках записаних по рядках без пропусків у рядку і між рядками;

 результат зберігається у файлі (Приклад 1) у вигляді:

- дані про місце розташування червоних шашок (в інших місцях знак “-“) ;

- порожній рядок;

- кількість чорних, білих, червоних і зелених шашок через пропуск;

якщо шашка якогось кольору відсутня на дошці то вивести у файл повідомлення BAD ІNPUT LІST (Приклад 2).

Приклад 0:

0 – колір і місце розташування ЧОРНОЇ шашки

1 – колір і місце розташування БІЛОЇ шашки

2 – колір і місце розташування ЧЕРВОНОЇ шашки

3 – колір і місце розташування ЗЕЛЕНОЇ шашки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 1 | 1 | 0 | 3 | 0 | 3 | 1 |
| 0 | 1 | 1 | 3 | 3 | 1 | 0 | 0 |
| 1 | 2 | 1 | 3 | 1 | 0 | 1 | 2 |
| 1 | 1 | 1 | 1 | 2 | 2 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 | 2 | 1 | 2 |
| 0 | 1 | 0 | 1 | 1 | 2 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 2 | 2 | 2 | 3 | 3 |

**Приклади**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| іnput.txt | output.txt |  | іnput.txt | output.txt |
| 82110303101133100121310121111221011101212010112110000000011122233 | 2----------------2-----2----22-------2-2-----2-------------222—18 28 11 7 |  | 82111313111133133121311121111221311121212111112111111311111122233 | BAD ІNPUT LІST |

|  |  |
| --- | --- |
| #include <cstdlib>#include "fstream"#include "string"using namespace std;int main(){ifstream cin("input.txt");ofstream cout("output.txt");string a[100000];long long int k=0;long long int k0,k1,k2,k3,i,n;k0=0;k1=0;k2=0;k3=0;cin>>n;while (! cin.eof()){k++;cin>>a[k];for (i=0;i<a[k].length();i++){if (a[k][i]=='0') k0++;if (a[k][i]=='1') k1++;if (a[k][i]=='2') k2++;if (a[k][i]=='3') k3++;if (a[k][i]=='1' || a[k][i]=='3' || a[k][i]=='0') a[k][i]='-';}}if (k0==0 || k1==0 || k2==0 || k3==0) cout<<"BAD INPUT LIST"<<endl;else{ for (i=1;i<k;i++) cout<<a[i]<<endl; cout<<endl; cout<<k0<<" "<<k1<<" "<<k2<<" "<<k3<<endl;} return 0;} | program Project2;{$APPTYPE CONSOLE}var ch,chr,b,z,i,n: integer; c: char; inp,outp: text;begin assign(inp,'input.txt'); assign(outp,'output.txt'); reset(inp); rewrite(outp); ch:=0; b:=0; chr:=0; z:=0;readln(inp,n) while not eof(inp)do begin while not eoln(inp)do begin read(inp,c); case c of '0': begin inc(ch); write(outp,'-')end; '1': begin inc(b); write(outp,'-')end; '2': begin inc(chr); write(outp,'2')end; '3': begin inc(z); write(outp,'-')end; end; end; writeln(outp); readln(inp); end; writeln(outp); if (ch=0)or(b=0)or(z=0)or(chr=0)then begin rewrite(outp); writeln(outp,'BAD INPUT LIST'); end else writeln(outp,ch,' ',b,' ',chr,' ',z); close(inp); close(outp);end. |

**Задача D. Цвяхи (100 балів)**

**Ім’я вхідного файлу: іnput.txt**

**Ім’я вхідного файлу: output.txt**

**Ліміт часу: 2с.**

На прямій дощечці вбиті цвяхи. Будь-які два цвяхи можна з'єднати ниткою. Потрібно з'єднати якісь пари цвяхів ниткою так, щоб до кожного цвяху була прив'язана хоча б одна нитка, а сумарна довжина всіх ниток була мінімальна.

**Вхідні дані**

У першому рядку вхідного файлу записано число N – кількість цвяхів (2 ≤ N ≤ 100). У наступному рядку записано N чисел -

координати всіх цвяхів (невід'ємні цілі числа, не перевищують 10000).

**Вихідні дані**

У вихідний файл потрібно вивести єдине число мінімальну сумарну довжину всіх ниточок.

**Приклад**

|  |  |
| --- | --- |
| іnput.txt | output.txt |
| 54 10 0 12 2 | 6 |

|  |  |
| --- | --- |
| #include <cstdlib>#include "fstream"#include "string"using namespace std;int main(){ifstream cin("input.txt");ofstream cout("output.txt");string a[100000];long long int k=0;long long int k0,k1,k2,k3,i;k0=0;k1=0;k2=0;k3=0;while (! cin.eof()){k++;cin>>a[k];for (i=0;i<a[k].length();i++){if (a[k][i]=='0') k0++;if (a[k][i]=='1') k1++;if (a[k][i]=='2') k2++;if (a[k][i]=='3') k3++;if (a[k][i]!='2') a[k][i]='-';}}if (k0==0 || k1==0 || k2==0 || k3==0) cout<<"BAD INPUT LIST"<<endl;else{ for (i=1;i<k;i++) cout<<a[i]<<endl; cout<<endl; cout<<k0<<" "<<k1<<" "<<k2<<" "<<k3<<endl;} return 0;} | function min(a,b:integer):integer;begin if a<b then min := a else min := b;end; procedure Swap(var a,b:integer);var q:integer;begin q := a; a := b; b := q;end; var n,i,j:integer; a,b:array[-1..100] of integer;begin assign(input,'input.txt'); reset(input); assign(output,'output.txt'); rewrite(output); Read(n); for i := 1 to n do Read(a[i]); for i := 1 to n do for j := i+1 to n do if a[i]>a[j] then Swap(a[i],a[j]); a[0] := Maxint; a[-1] := 0; for i := 1 to n do b[i] := min(b[i-1],b[i-2])+abs(a[i]-a[i-1]); WriteLn(b[n]); close(input); close(output);end. |

**Задача E. Міста (100 балів)**

**Ім’я вхідного файлу: іnput.txt**

**Ім’я вхідного файлу: output.txt**

**Ліміт часу: 5с.**

У файлі мститься N назв міст (по одній назві в кожному рядку). Утворіть з даного набору слів замкнений ланцюжок, в якому кожне наступне слово починається з літери, якою закінчувалось попереднє, використавши найбільшу кількість слів. Всі слова у файлі різні і у ланцюжку їх можна використовувати не більше одного разу. Програма повинна на екран та у перший рядок файлу вивести кількість використаних слів,.У випадку, коли ланцюжок утворити неможливо, у файлі міститься лише одне число 0.

**Вхідні дані**

У першому рядку вхідного файлу записано число N – кількість цвяхів (2 ≤ N ≤ 50). У наступному рядку записано N слів великими латинськими літерами, які задають імена міст.

**Вихідні дані**

У вихідний файл потрібно вивести єдине число максимальну кількість міст.

**Приклади**

|  |  |
| --- | --- |
| іnput.txt | output.txt |
| 10KYEVDONETSKLUTSKKOVELVINNICADNIPROPETROVSKKIROVOGRADLUGANSKIVANO-FRANKOVSKVINNICA | 4 |

|  |  |
| --- | --- |
| #include <fstream>#include <algorithm>#include <string.h>using namespace std;ifstream cin("input.txt");ofstream cout("output.txt");struct ch{        char f;        char l;};int m[52][52];int used[52], maxStep;void p(int step, int n, int v, int fv){        used[v] = 1;        int i;        for (i = 0; i < n; i++)        {                if (m[v][i] == 1)                {                        if (i == fv)                        {                                maxStep = max(maxStep, step);                        }                        else                        {                                if (used[i] != 1)                                {                                        p(step + 1, n, i, fv);                                }                        }                }        }        used[v] = 0;}int main(){        int n,i,j;        ch mm[50];        char a[100];        cin >> n;        for (i = 0; i < n; i++)        {                cin >> a;                mm[i].f = a[0];                mm[i].l = a[strlen(a) - 1];        }        for (i = 0; i < n; i++)        {                for (j = 0; j < n; j++)                {                        if (mm[i].l == mm[j].f)                        {                                m[i][j] = 1;                        }                }        }        for (i = 0; i < n; i++)        {                p(1, n, i, i);        }        cout << maxStep << "\n";        return 0;} | program state;const MaxState = 50;var name : array [1..MaxState] of string[25]; { назви міст }fl : array [1..MaxState] of boolean;kol, maxkol : array [1..Maxstate] of byte;n, max, j, k, i : byte;f : text;flag : boolean;{ перевірка можливості додати наступне слово }function next (st1, st2: string) : boolean;beginif st1[length(st1)] = st2[1] then next := trueelse next := false;end;begin { введення даних з файлу }assign(f,'input.txt'); reset(f); readln(f,n);n := 0;repeatinc(n); readln(f, Name[n]);until eof(f);close(f); { головний алгоритм, який шукає тільки один найдовший ланцюжок }flag := false;for i:=1 to n do fl[i] := true;j := 0; k := 0; max := 0;repeatwhile (k < n) dobegininc(k);if (j = 0) or (next(Name[kol[j]], Name[k]) and fl[k]) thenbegininc(j); kol[j] := k;fl[k] := false;k := 0;end;end;if (j > max) and next(Name[kol[j]], Name[kol[1]]) thenbeginmax := j;maxkol := kol;end;if j>0 thenbegink := kol[j]; dec(j);fl[k] := true;endelse flag := true;until flag; { виведення результатів роботи одночасно на екран і у файл }assign(f, 'output.txtl'); rewrite(f);if max = 0 then writeln(f,'0') elsebeginwriteln(f, max);end;close(f);end. |