

Sfaturi de bună practică

pentru concurenții OJI / ONI 2012

Elevii care vor participa la Olimpiada Județeană/Națională de Informatică trebuie să aibă în vedere următoarele:

- A) Pentru evitarea erorilor la compilare, codul surselor trebuie să **respecte Standardul C++**, respectiv **Standardul Free Pascal** (cu alte cuvinte, să fie corect din punct de vedere al limbajului)
- B) Faptul că vitezele de citire/scriere a datelor pot să difere de la o distribuție GNU C la alta.
- C) Faptul că există reguli de formatare a fișierelor de test la concursurile de algoritmică.

A) Respectarea standardului limbajului.

Codul scris în conformitate cu standardul limbajului este un **cod portabil** și va fi compilat fără erori cu oricare compilator GNU C.

Ne vom referi în special la limbajele C/C++, deoarece în cazul acestora apar cele mai multe situații de practică defectuoasă.

1. **Nu salvați soluțiile cu extensia .c, decât dacă sunteți siguri că NU doriți să utilizați biblioteca C++.**

Fișierele cu extensia **.cpp** vor fi compilate conform standardului C++ (ISO C++ 98), iar cele cu extensia **.c** vor fi compilate conform standardului C (ISO C 99). În exemplu, sursa. c nu va compila, în timp ce sursa. cpp compilează:

sursa.c	sursa.cpp
<pre>// Se foloseste biblioteca C #include <stdio.h> // OK // NU puteti folosi Biblioteca C++ !! #include <fstream.h> // Eroare struct A { }; A x; // Eroare (standardul C) int main() { // variabila locala i in for for (int i = 0; i < 10; ++i) //Eroare printf("%d", i); return 0; }</pre>	<pre>// Se foloseste biblioteca C #include <stdio.h> // OK // Se foloseste Biblioteca C++ #include <fstream.h> // OK struct A { }; A x; // OK (standardul C++) int main() { // variabila locala i in for for (int i = 0; i < 10; ++i) // OK printf("%d", i); return 0; }</pre>

1. Header-e

Standardul limbajul C++ definește 33 de headere, conform tabelului:

<algorithm>	<iomanip>	<list>	<queue>	<streambuf>
<bitset>	<ios>	<locale>	<set>	<string>
<complex>	<iosfwd>	<map>	<sstream>	<typeinfo>
<deque>	<iostream>	<memory>	<stack>	<utility>
<exception>	<istream>	<new>	<stdexcept>	<valarray>
<fstream>	<iterator>	<numeric>	<sstream>	<vector>
<functional>	<limits>	<ostream>		

Facilitățile **Bibliotecii C Standard** sunt furnizate de 18 headere adiționale:

<cassert>	<ciso646>	<csetjmp>	<cstdio>	<ctime>
<cctype>	<climits>	<csignal>	<cstdlib>	<wchar>
<cerrno>	<locale>	<cstdlib>	<cstring>	<wctype>
<cfloat>	<cmath>	<cstddef>		

IMPORTANT!

Heder-ele cu extensia .h sunt *deprecated* pentru limbajul C++. Aceasta înseamnă că deși unele compilatoare (cum sunt și cele de la OJI/ONI 2011) mai suportă stilul vechi de declarare, compilatoarele de ultimă generație nu-l mai acceptă!

Exemple:

Corect	Deprecated (pentru headere C++)
<pre>#include <fstream> // header-e C++ #include <iostream> #include <iomanip> #include <cmath> // header-e C #include <cstdio> #include <cstring> using namespace std; // directiva using precizează spațiul de nume std // în care este definită Biblioteca Standard C++</pre>	<pre>// deprecated #include <fstream.h> #include <iostream.h> #include <iomanip.h> // Corect (nondeprecated) #include <math.h> #include <stdio.h> #include <string.h></pre>

Observație: Headerele C, standardul acceptă (deocamdată) declarații cu extensia .h : <stdio.h>, <math.h>, <string.h>, etc.

B). Vitezele operațiilor de intrare-ieșire în limbajele C/C++

Pentru o distribuție GNU C dată, operațiile de citire/scriere C și C++ diferă uneori în ce privește viteza de executare.

Funcțiile `scanf()`, `printf()` de pildă, sunt pentru anumite distribuții, mai rapide decât operațiile de inserție (`<<`) sau extracție (`>>`) din stream-uri, în timp ce pentru alte distribuții lucrurile stau exact invers !

Concurenții sunt sfătuiți să studieze mediile de lucru pentru OJI/ONI și să aleagă acele metode de citire/scriere pe care le consideră optime.

1. OJI 2012.

Compilatorul mediului MinGW Developer Studio are particularitatea că produce executabile pentru care **vitezele de citire-scriere cu fișiere sunt mai rapide decât operațiile similare cu stream-uri.**

2. ONI 2012

Pentru mediile instalate la ONI 2011, (Ubuntu 8.04: **gcc 4.4.2**, Windows XP: **gcc 3.3.1**), s-au facut teste de viteză, așa cum se poate vedea în tabelul de mai jos:

Ubuntu 8.04

gcc 4.2.4

Procesor: Intel Dual CoreE2200, @2.20GHz/ Core

Iteratii	<stdio>		<fstream>	
	fscanf()	fprintf()	>>	<<
1000000	0,240 sec	0.170 sec	0,19 sec	0.29 sec
10000000	2.35 sec	1.85 sec	1.65 sec	3.15 sec

S-a rulat fiecare caz de zece ori și s-a luat media.

Compilare: **gcc/g++ -lm -static numfis.cpp**

Concluzie:

E mai rapidă citirea din streamuri cu operatorul >> și scrierea cu funcția fprintf().

Windows XP SP2

gcc 3.3.1 (pachetul OJI)

Procesor: Intel Dual CPU T2370, @1.73GHz/Core

Iterații	<stdio>		<fstream>	
	fscanf()	fprintf()	>>	<<
1000000	0,51 sec	0.60 sec	3,72 sec	0.65 sec
10000000	5.47 sec	6.95 sec	35.2 sec	7.10 sec

S-a rulat fiecare caz de zece ori și s-a luat media.

Compilare: **gcc/g++ numfis.cpp**

Concluzie: Dacă scrierea se face cu viteze aproximativ egale, în schimb, în ce privește citirea, **fscanf()** e mult mai rapidă decât operația de citire a datelor din stream-ul de intrare cu operatorul **>>**.

ATENȚIE! **NU folosiți endl . Utilizați ' \n ' .**

La scrierea în streamuri, datele de ieșire se acumulează într-un buffer (stream) care se golește periodic, nefiind nevoie de accesarea discului la fiecare operație de scriere.

Manipulatorul de format **endl**, face *flush* le stream-ul de ieșire, ceea ce forțează scrierea pe disc. Dacă aceasta se întâmplă într-un ciclu, atunci viteza scade catastrofal:

Ubuntu 8.04,

gcc 4.2.4,

AMD Athlon(TM) XP 2500+

<pre>#include <fstream> using namespace std; int main() { ofstream fout("numere.in"); int n = 10000000; for (int i = 0; i < n; ++ i) fout << 7 << '\n'; fout.close(); }</pre>	<pre>#include <fstream> using namespace std; int main() { ofstream fout("numere.in"); int n = 10000000; for (int i = 0; i < n; ++ i) fout << 7 << endl; fout.close(); }</pre>
Timp de executare: 3,21 secunde	Timp de executare: 40.5 secunde !!!!

C) Formatul fișierelor

1. La OJI și ONI, ca de altfel la toate competițiile de algoritmică naționale sau internaționale, este o practică curentă aceea ca **ultima linie din fișierele de test de intrare, cât și din cele de ieșire, să se termine cu caracterul newline**. Concurenții trebuie să țină seama de acest lucru.
2. **Salvați sursele doar cu numele și extensiile enumerate în enunțul problemelor.**
3. **Respectați formatul fișierului de de ieșire!**
De exemplu, fie o problemă care are două cerințe: a) și b). Să presupunem că se cere ca răspunsul pentru cerința a) trebuie se găsească pe linia 1, iar răspunsul pentru cerința b) să se găsească pe linia 2. În situația în care nu ați reușit să rezolvați cerința a) dar aveți un răspuns pentru b), veți scrie răspunsul pentru cerința b) pe linia 2 și nu pe prima linie !