**Az alapfeladat:**

Mint az a wikin is olvasható, a feladat egy primitív fordítóprogram összeállítása, ami egy fiktív „Aritmetika” nevű nyelvben írt kódot lebont egyszerű, egy- vagy kétoperandusos műveletekre. Ez megtévesztően bonyolultan hangzik, ám maga a feladat csupán annyi, hogy a program kérjen be egy szöveget (fájlból, konzolból, wx-es TextCtrl-ról, stb.), majd ezt fordítsa át gépi kódra, és ezt írja ki egy szövegfájlba (vagy amibe szeretnétek).

Ilyen szempontból az alapfeladat csak annyi, hogy a program kérjen be egy szöveget, és abból gyártson egy másik, az eredetivel azonos értelmű szöveget. Ha az Aritmetika program már lefordul, és fut a generált kód, akkor az alapfeladat kész van, függetlenül attól, hogy mennyire optimális. (Viszont ez valószínűleg nem fog sok pontot érni. Nem ismerem a pontozást, de én kb. 50, max 60%-ra tippelnék a feladat nehézségéből.)

Az alapfeladatnak tudnia kell:

1. A négy alapműveletet (+,-,\*,/) használnia.
2. Tetszőlegesen zárójelezhetőnek kell lennie. (Akár többszörös, egymásba ágyazott zárójelekkel is! Ez azt jelenti, hogy a precedencia szabályok implementálásakor hibás azt feltételezni, hogy a műveleti hierarchiának csak három szintje lehet!)
3. Képesnek kell lennie kiszűrni a fordításidejű hibákat. (Lásd: Wiki)
4. Helyes eredményeket kell tudnia adni. (lebegőpontos pontatlanságok mellett)

Az alapfeladatnak NEM kell tudnia:

1. Wx-et használni. A programnak csak fordítania kell tudni, az teljesen mindegy, hogy milyen primitív/fancy UI-t használtok. Ugyanakkor viszont semmi nem is tiltja, hogy Wx-et használjatok. Flugi azt tanácsolta, hogy minél könnyebb szerkeszteni a kódotokat, annál egyszerűbb a debugolás, ezért nem rossz ötlet kicsit több időt eltölteni a UI-al, hogy a későbbiekben ne kelljen sokat szenvedni. Emellett, bár nem jár plusz pont a szép UI-ért, valószínűleg nem hátrány, ha látszik, hogy fektettetek bele energiát.
2. Speciális műveleteket (SIMD, FMA3) használnia. **Ez valószínűleg nagy pontlevonásokkal jár, ha nem működik, úgyhogy szerintem azért mindenki próbálja meg megcsinálni.**

**A további feladatok:**

1. Optimalizálás. Az egyik legfőbb tényező, ami alapján a pontozás el lesz döntve, az ez. (A különleges műveletek is elsősorban ebben segítenek.) A legfontosabb optimalizálási szempont a sorok száma. A futásidő, processzor- és memóriahasználat, stb. szinte egyáltalán nem számítanak, csak az, hogy minél kevesebb sorból álljon a gépi kód.
2. Regiszterek használata memória helyett. Bár a mi feladatunkban ez nem látszik, a való életben a regiszterek elérése gyorsabb, mint a memóriáé. Bár valószínűleg nem túl sokat, de egy pár pontot bizonyára ér, ha inkább regiszterekkel dolgozik a program. (A regiszterek száma azonban, a memóriával ellentétben véges!)
3. Speciális műveletek:
   1. SIMD: „Párhuzamos” értékadás. Ha két azonos művelet egy sorban van, akkor elméletben egyszerre mennek végbe. Ezzel jelentősen le lehet rövidíteni a kódot, ám az egyszerre elvégezhető műveletek száma korlátozott. Ezen felül csak azonos fajta műveleteket lehet összevonni, tehát pl. szorzás és összeadás nem szerepelhet egy sorban. Ezen felül nem adhattok értéket egy regiszternek/memóriának, és használhatjátok fel azt egy lépésben. (lásd: wiki)
   2. FMA3: Ez egy olyan művelet, ami elvégez egy szorzást két operanduson, majd hozzáadja a baloldalon álló regiszter értékéhez. (Lényegében ugyanúgy működik, mint ahogy C++-ban működne.)

A SIMD-nek az FMA3-al is kell tudnia működni!

**Példa lefordított programra:**

**ARITMETIKA:**

3 2   
KIA=bea+3+10\*beb+4+15;  
KIB =3+bea+(9\*(4+5));

Az első két szám a felhasználható regiszterek száma és a párhuzamos műveletek száma. Ha csak egy szám van, akkor a két szám egyenlő. (Ha nincs egy szám sem, akkor a programnak valószínűleg vagy hibát kell dobnia, vagy alapértelmezett értékekkel kell futnia.)

**GÉPI KÓD:**

Reg[0]=bea+3;  
Reg[1]=10\*beb;  
Reg[2]=Reg[0]+Reg[1]; Mem[0]=4+15;  
KIA=Reg[2]+Mem[0];  
Reg[1]=4+5;  
Reg[0]+=9\*Reg[1];  
KIB=Reg[0];

Érdemes észrevenni, hogy mivel csak három regiszter áll rendelkezésünkre, ezért a Mem[0]=4+15; már memóriába tárolódik.

**Általános tanácsok:**

1. Vannak nagyon tipikus dolgok, amikkel sok sort lehet spórolni. Itt egyértelműen nem írhatom le, hogy ezek mik, de ha ránéztek a kódra, már azzal is könnyen rá lehet jönni sok dologra.
2. Ne kezdjetek neki addig a programozásnak, amíg nem tudjátok legalább nagy vonalakban, hogy hogyan akarjátok megírni a programot. Az „egyelőre nincs ötletünk, ezért elkezdjük, és hátha működik” egy nagyon káros hozzáállás, ugyanis ha rossz az alapötlet, akkor lehet, hogy egyszerűbb elölről kezdeni, mint átírni. Az sem káros, ha előre tudjátok, hogy ki pontosan mit fog megírni.
3. Általában a gyakorlaton elhangzott anyagok nem véletlenül most kerültek elő. A legtöbb esetben a gyakorlati anyagoknak legalább érintőlegesen (de inkább szorosabban) köze van a nagybeadandóhoz.
4. Ha nincs ötletetek egy bizonyos témával (pl. optimalizálás) kapcsolatban, akkor némi google/wiki búvárkodás rengeteget tud segíteni. Bár sajnos konkrét segítséget nem tudok adni, annyit talán mondhatok (mivel valamelyik gyakorlaton tudtommal elhangzott), hogy a „programkód értelmezése” műveletet tisztes nevén „parsing”-nak hívják.
5. A határidő Jan. 5, a karácsonyi szünet utáni első péntek. Aki még ezen a napon bemutatja a programját, az kaphat max pontot. Ha valakiknek kell idő, akkor (elméletben) a vizsgaidőszak végéig ki lehet húzni a határidőt, ám a késés mértékének függvényében a csapatok pontlevonással kezdenek.

**Egyelőre nem ismert/tilos dolgok:**

1. Léteznek a C++-ban parser generáló std könyvtárak. Ezeket értelemszerűen tilos használni. (És Boost könyvtárakkal sem trükközünk!)
2. Egyelőre nem tudom, hogy a bemeneteket hol kell megadni. ~~Ezzel nekünk valószínűleg nem kell foglalkoznunk, mert ez vélhetőleg az interpreter része lesz.~~

A bemeneteket a programnak valahonnan (konzolból, fájl részeként, stb.) be kell kérnie.

1. ~~Egyelőre nem tudom, hogy lehet-e regisztereket felülírni, ezt meg fogom kérdezni~~.

A memóriákat és regisztereket felül lehet írni, csak a kimeneteket nem.

Ha bárkinek kérdése van, akkor megpróbálok minél előbb válaszolni. Ha én nem tudom a választ, akkor minden hétfőn és csütörtökön találkozok Flugival, leghamarabb ezeken a napokon tudok kérdezni tőle.