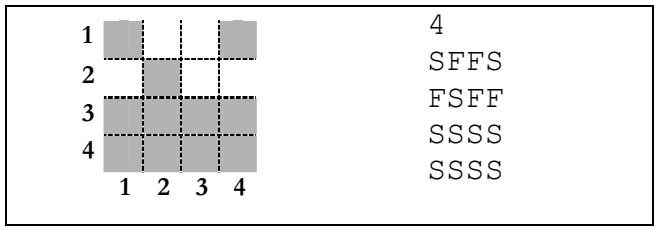
Feladat

A fraktál-tömörítés elvét kell az alábbi feladatot megoldó programba beépíteni. Az egyszerűség kedvéért egy négyzetes pixel-grafikus kép szöveggé kódolását kell elvégezni.

Bemenet (text-fájl: `kep.be`):

1. sor: N, a kép mérete, azaz pixel-sorok és –oszlopok száma, 2-hatvány ($N=2^k$); a példában 4.

2-N+1. sor: a kép 1-N. pixel-sora, amelyek pontosan N darab, színeket kódoló jelből áll ('A'..'Z'); a példában előforduló két színt az 'S' és az 'F' betű kódolja.

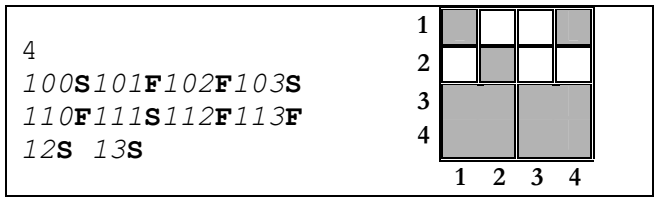


Egy 4x4-es példa-kép… … és fájlbeli megfelelője.

1. ábra

Kimenet (text-fájl: `kep.ki`):

1. sor: N, a kép mérete; a példában 4.
2. sor: a kép egészét kódoló *prefix*+**szín** párokból álló szöveg, amelyet csak a könnyebb értelmezhetőség kedvéért tagoltuk: a négy szövegdarab a magyarázó ábra 4 blokkjának felel meg.



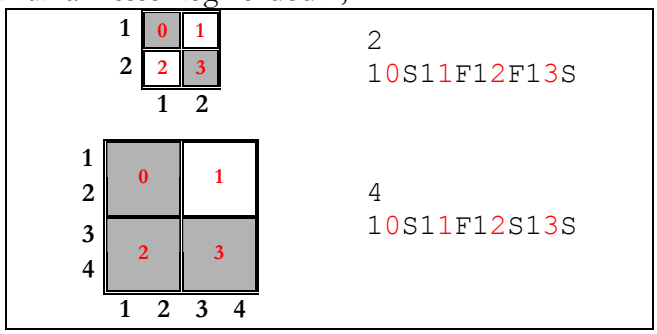
A fenti kimenete, valamint a magyarázat.

2. ábra

A **fraktál-kódolás lényege** a következő:

1. egy homogén, S-sel jelölt színű, bármekkora oldalhosszúságú kép kódja: `0S` (=nulla jel + S betű);
2. a nem egyszínű képet 4, egybevágó részképre bontjuk (ezért az $N=2^k$ feltétel);
3. a bontás tényét az jelzi, hogy a képelemek prefixuma 1-essel fog kezdődni;
4. a bal-felső sarokbeli részkép minden képelemének prefixuma: 0,
5. a jobb-felsőé: 1,
6. a bal-alsóé: 2,
7. a jobb-alsóé: 3.

A példákból látszik, hogy a „makroszkopikus" (fraktál-) struktúra tükröződik a prefixumokon, az egyes képrészek tényleges méretét a kimenet 1. sora, valamint a felezések számát tükröző prefixum-hossz határozza meg. Most is a tagolás az értelmezést hivatott segíteni. (Nézze meg ismét a 2. ábra példáját!)



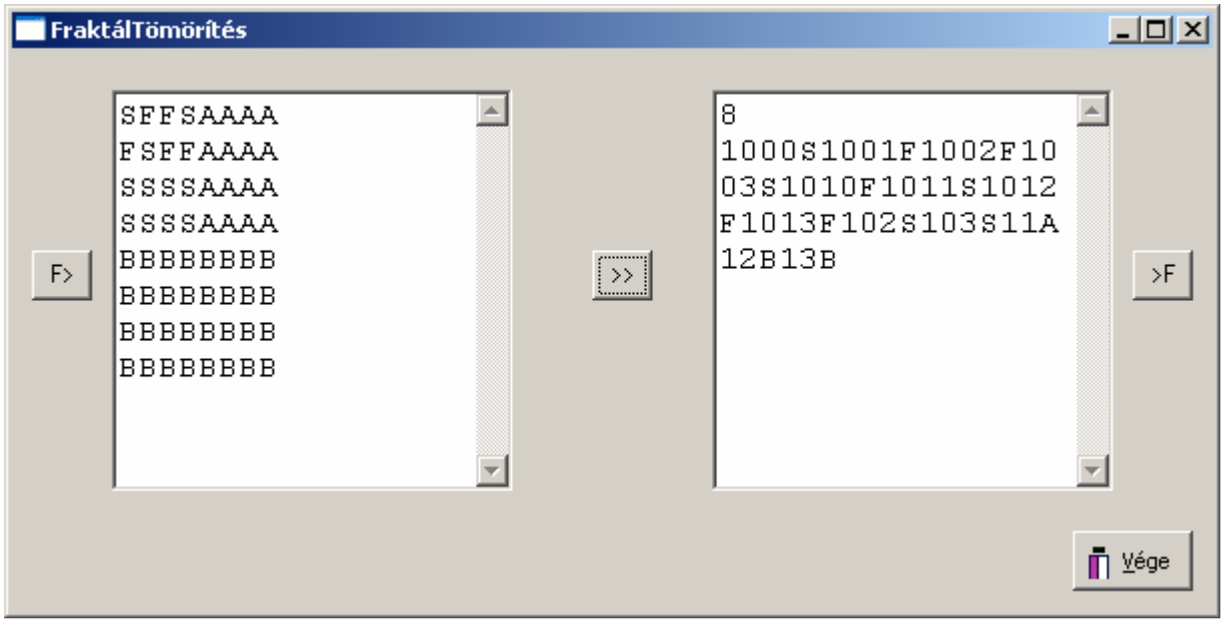
Példa-képek… … és leírásuk.

3. ábra

Feladata olyan alkalmazás írása, amely

a. beolvassa a felhasználó által megadott nevű (`*.be`), „szintaktikusan" helyes fájlból a képet,
b. megjeleníti a színmátrixot valamilyen erre alkalmas grafikus komponensben (stringGrid, memo), azaz a fájl tartalmát az első sora nélkül,
c. elvégzi a fenti fraktál-kódolást,
d. megjeleníti az eredményt (a kimenet mindkét sorát) egy erre alkalmas grafikus komponensben (stringGrid, memo),
e. kimenti egy '`.ki`' kiterjesztésű, a fent leírt szerkezetű text-fájlba (a fájl „vezeték-neve" megegyezik a beolvasottéval);
f. módot biztosít, hogy a bementi grafikus komponensben „kézzel" is módosításokat tehessen a felhasználó (a szintaktikus helyesség megőrzése mellett),
g. gombokkal vagy menüvel vezérelhető a program: fájlkezelés, kódolás, kilépés.

Egy GUI példa:



Pontozás:

Tevékenység	Pont	Tevékenység	Pont
Minden funkcióhoz grafikus kellék a formon	6	Az input szerkeszthető kézzel (és feldolgozható)	2
Fájl betölthető az input komponensbe	2	Az eredmény (bármi) fájlba vihető	1
Valami átkerül az input komponensből az outba	1	Helyes az eredmény a fájlban	1
Eredmény OK az 1. próbafájlra (fájlban/ablakban)	3	Helyes az output első sora	1
Eredmény OK a 2. próbafájlra (fájlban/ablakban)	5	A funkciók elérhetősége kezdetben megfelelő	1
Eredmény OK a 3. próbafájlra (fájlban/ablakban)	7	A funkciók elérhetősége jól módosul betöltés után	1
Eredmény OK a 4. próbafájlra (fájlban/ablakban)	7	Pluszok	4
Összesen:	**41**		
Értékelés			
Kettes, ha legalább:	13	Négyes, ha legalább:	27
Hármas, ha legalább:	20	Ötös, ha legalább:	34