NEUMANN JÁNOS EGYETEM

Programozó Informatikus Szakirányú Továbbképzés

Programozás Beadandó Feladat

Borics Krisztián

2025.10.20

**RGB színek**

Fényképek számítógépes tárolására sokféle képformátumot ismerünk. A legegyszerűbb típusok az eredeti kép pixeleihez tartozó színértékeket tárolják, például ilyen a BMP vagy a fényképezőgépek által használt RAW formátum. Az ilyen képállomány bináris, melynek első néhány bájtja a kép jellemzőit írja le, míg a további bájtok sorfolytonosan a kép egy-egy pixelének színét adják meg.

Ebben a feladatban egy RAW formátumú fájlból átalakított szöveges állományt kell feldolgoznia. A kep.txt szöveges állomány egy 640×360 méretű, RGB kódolású képet ír le. Az állomány csak a képpontok színét tartalmazza sorfolytonosan, azaz a fájl 360 sorának mindegyike 640 képpontból, képpontonként három színértékből áll. Az első szám a piros (Red), a második szám a zöld (Green) és a harmadik szám a kék (Blue) értéket tartalmazza. Az RGB értékek 0 és 255 közötti egészek, melyeket a fájlban egy-egy szóköz választ el. Részlet az állomány első néhány sorából (az alábbi példákban szereplő számhármasok félkövér betűstílussal vannak kiemelve):

0 85 112 0 86 113 0 86 113 0 86 113 0 86 113 0 86 113 …

0 86 111 0 86 111 0 86 113 0 86 113 0 86 113 0 86 113 …

0 86 113 0 86 113 0 86 113 0 86 113 0 86 113 1 87 114 …

0 86 113 0 86 113 0 86 113 0 86 113 1 87 114 1 87 114 … …

Az első sor első három száma a kép bal felső képpontjának színe, azaz RGB(0, 85, 112), míg a második sor harmadik pixelének színe RGB(0, 86, 113), a negyedik sor hatodik oszlopában lévő képpont színe RGB(1, 87, 114). A szöveges állomány által leírt kép:

A képen ég, felhő, felhők, képernyőkép látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

Készítsen programot, amely az állomány adatait felhasználva megoldja az alábbi feladatokat! A program forráskódját mentse rgb néven! A program megírásakor a felhasználó által megadott adatok helyességét, érvényességét nem kell ellenőriznie, és feltételezheti, hogy a beolvasandó adatok a leírtaknak megfelelnek.

A képernyőre írást igénylő részfeladatok esetén – a mintához tartalmában hasonlóan – írja ki a képernyőre a feladat sorszámát (például: 2. feladat:), és utaljon a kiírt tartalomra is! Ha a felhasználótól kér be adatot, jelenítse meg a képernyőn, hogy milyen értéket vár! Mindkét esetben az ékezetmentes kiírás is elfogadott.

1. Olvassa be a kep.txt állomány tartalmát, és tárolja el a 640×360 képpont színét!

2. Kérje be a felhasználótól a kép egy pontjának sor- és oszlopszámát (a számozás mindkét esetben 1-től indul), és írja a képernyőre az adott képpont RGB színösszetevőit a minta szerint!

3. Világosnak tekintjük az olyan képpontot, amely RGB-értékeinek összege 600-nál nagyobb. Számolja meg és írja ki, hogy a teljes képen hány világos képpont van!

4. A kép legsötétebb pontjainak azokat a pontokat tekintjük, amelyek RGB-értékeinek összege a legkisebb. Adja meg, hogy mennyi a legkisebb összeg, illetve keresse meg az ilyen RGB összegű pixeleket, és írja ki mindegyik színét RGB(r,g,b) formában a mintának megfelelően!

5. A képen a kék ég látható közepén egy felhővel. Az ég és a felhő színe között jelentős különbség van, így az ég-felhő határvonal programmal is felismerhető. Ennek megtalálásához készítsen függvényt határ néven, amely megadja, hogy egy adott sorban van-e olyan hely a képen, ahol az egymás melletti képpontok kék színösszetevőinek eltérése meghalad egy adott értéket! A függvény kapja meg paraméterként a sor számát, illetve az eltérés értékét, melyek egészek! A függvény visszatérési értéke egy logikai érték legyen, amely megadja, hogy az adott sorban volt-e az eltérést meghaladó különbség az egymás melletti képpontok kék színében!

6. Keresse meg a képen a felhő első és utolsó sorát az előzőleg elkészített függvény segítségével úgy, hogy eltérésként 10-et ad meg a függvénynek bemenetként! Adja meg az első és az utolsó olyan sor sorszámát, ahol az eltérés a soron belül valahol 10-nél nagyobb!

**Bemutatás**

A feladat megvalósításához C# nyelvet választottam. A user interface megvalósítására WPF – MVVM technológiát használtam. A program nem igényel beállításokat, kicsomagolás után futtatható, ha van .NET 8 telepítve a gépre. A szöveg file importálása a megnyitás gombbal lehetséges.

A képen szöveg, képernyőkép, felhő, szoftver látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

A feladat kiírásnak megfelelően a funkciók:

A felhasználó által megadott koordináta színe:

A képen szöveg, képernyőkép, szoftver, Operációs rendszer látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

A koordináták megadása után a Grab color gombbal lehet lekérdezni a szín komponenseit.

Világos képpontok száma

A light pixel gombra kattintva számolja meg azon pixelek számát, amelyeknek az összege > 600.

A képen szöveg, képernyőkép, szoftver, Operációs rendszer látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

A felhő alsó és felső határai

A borders gombra kattintva kapjuk meg a felhő alsó és felső határait.

A képen szöveg, képernyőkép, multimédia, szoftver látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

Sötét pixelek

A dark pixel gombra kattintva kapjuk meg a legsötétebb pixeleket és az ehhez tartozó minimum értéket a jobb oldali listában.

A képen szöveg, képernyőkép, felhő, szoftver látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

**Kidolgozás**

A project struktúra a lenti képen látható.

A képen szöveg, elektronika, képernyőkép, szoftver látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

Model

Az ImagePoint nevű adatstruktúra helye. A megnyitott képet List<ImagePoint> listába helyezem bele. Az adatstruktúra szerkezete:

public class ImagePoint

{

public int sum => R + G + B;

public int avg => sum / 3;

public bool isLight => sum > 600;

public byte R { get; set; }

public byte G { get; set; }

public byte B { get; set; }

public int PointX { get; set; }

public int PointY { get; set; }

}

A struktúrát úgy építettem fel, hogy importáláskor számítsa ki a képpont összegét, átlagát és azt, hogy világosnak számít-e az adott képpont. A színkomponensek adatai mellett tárolva van a koordináta is. Ebből a szerkezetből LINQ segítségével könnyű információkat kinyerni, valamint könnyen képezhető Dictionary adattípus, mely gyors elérést biztosít az adatokhoz.

Classes

A képen szöveg, képernyőkép, szoftver, Betűtípus látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

A képkezelő osztálya. Ebben kapott helyet minden megvalósítandó feladathoz kapcsolódó metódus. Az osztályt ImagePoint típusú listával kell példányosítani. Az OpenFile statikus metódus végzi az importot, ezt el lehet érni példányosítás nélkül is.

img

Ez tartalmazza a használt erőforrásokat.

RelayCommand

A képen szöveg, képernyőkép, Betűtípus látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

Az ICommand interfészt a WPF binding használja a gombok, menük, stb. parancsok kezelésére. A „ProjectRelayCommand” egy egyszerű implementáció, amely egy Action-t és egy Func<bool>-t (vagy paraméteres változatban Action<object> és Func<object,bool> ) tárol, és ezeket hívja Execute illetve CanExecute során.

CanExecute határozza meg, hogy a parancs végrehajtható-e (a WPF ennek alapján engedélyezi / tiltja a gombot).

CanExecuteChanged esemény jelzi a binding rendszernek, hogy re-evaluate-olja a CanExecute állapotot.

ViewModel

Az MVVM tervezési model megvalósítása. Ez a model segít az üzleti logika és a nézetmodel elválasztásában. Bővebb info : [Modell–View-ViewModel - .NET | Microsoft Learn](https://learn.microsoft.com/hu-hu/dotnet/architecture/maui/mvvm)

Minden egyes a projectben résztvevő view-hoz kell készíteni egy view modelt. Esetünkben ez a MainWindow és a MainWindowViewModel páros. A MainWindod tartalmazza a nézetet XAML leírónyelven. DataContext-ként hozzá kell kapcsolni a view modelt a lentiek szerint:

<Window.DataContext>

<ViewModels:MainWindowViewModel/>

</Window.DataContext>

Így a view csak a nézetet, a ViewModel pedig a BL-t tartalmazza. Fontos, hogy a ViewModelnek tartalmazni kell egy INotifyPropertyChanged interfész implementációt, segítségével kommunikálhatunk a WPF-en , azaz a nézeten elhelyezett vezérlőkkel.