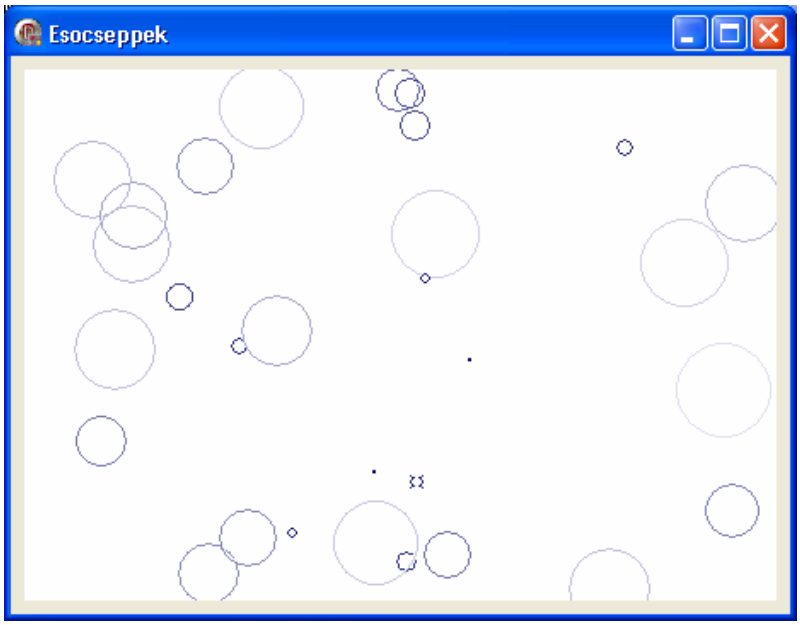
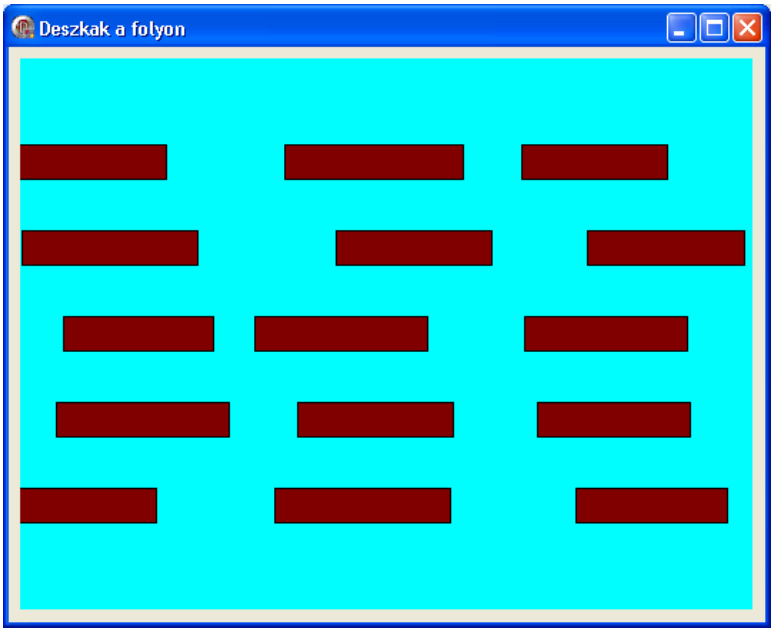
Gyakorlatok:

1. Készítsünk egy **TCsepp** osztályt, amely egy esőcseppet (kört) rajzol ki a megadott Image komponensre. A TCsepp osztálynak írjuk meg a **konstruktorát**, mely csak egy paramétert tartalmazzon – annak az Image (*ExtCtrls unitban található*) komponensnek a nevét, ahová az esőcseppet ki akarjuk rajzolni. A konstruktor generáljon ki egy véletlenszerű koordinátát ezen a képen (Image-en) és egy véletlen sugarat (0-tól 29-ig). Majd rajzolja ki az esőcseppet erre a koordinátára. Az osztály tartalmazzon még egy **kirajzol** és egy **letöröl** eljárást, amely kirajzolja a kört az objektumhoz tartozó koordinátára az objektumhoz tartozó sugárral. A kirajzolást bsClear (*graphics unitban található*) ecsetstílussal és RGB(sugár*8, sugár*8, 100+sugár*5) (*windows unitban található*) kék színárnyalatú körvonallal végezzük. Az osztálynak legyen még egy **növekszik** metódusa, amely letörli az esőcseppet, növeli a sugarát, majd megnézi hogy a sugár nem érte-e el a 30 pixelt. Ha elérte, akkor beállítja 0-ra és új koordinátákat generál ki az objektumnak. Végül kirajzolja az esőcseppet.

   Ezt az osztályt felhasználva készítsünk egy programot, amely megjelenít 30 esőcseppet és 25 század-másodpercenként növeli azok nagyságát (amíg nem érik el a 30-at, utána egy másik helyen 0 sugárral jelennek meg).

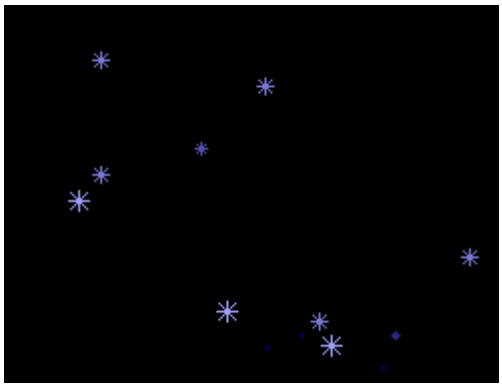


2. Készítsünk egy **TDeszka** osztályt, majd ennek segítségével azt az alkalmazást, melyben véletlen hosszúságú deszkák úszkálnak a vízen. Az első sor balra, a második jobbra, a harmadik megint balra, a negyedik megint jobbra, az ötödik sor pedig ismét balra ússzon. Ügyeljünk arra, hogy két egymás melletti deszka között mindig legyen egy kis hely. Az egyik szélén kiúszó deszkák a másik szélén ússzanak be (ne hirtelen jelenjenek meg).



3. Készítsünk **TCsillag** osztályt, amely kirajzol egy véletlen méretű (pl. 1-től 5-ig) csillagot. Az osztálynak legyen egy olyan metódusa, melynek meghívásával a csillag eggyel nagyobb méretű és világosabb kék színű lesz. Ha eléri az 5-ös méretet, akkor tűnjön el és egy új, véletlen helyen jelenjen meg 1-es mérettel és sötétkék színnel.

   A TCsillag osztály segítségével készítsünk egy képernyővédőt, amely teljes képernyőn kirajzol 100 csillagot, majd ezek méretét és fényességét a megírt metódus segítségével növeli (ha valamelyik csillag eléri a maximum méretét, akkor egy másik helyen jelenjen meg kis méretben). Az alkalmazás bármelyik billentyű megnyomásával fejeződjön be.

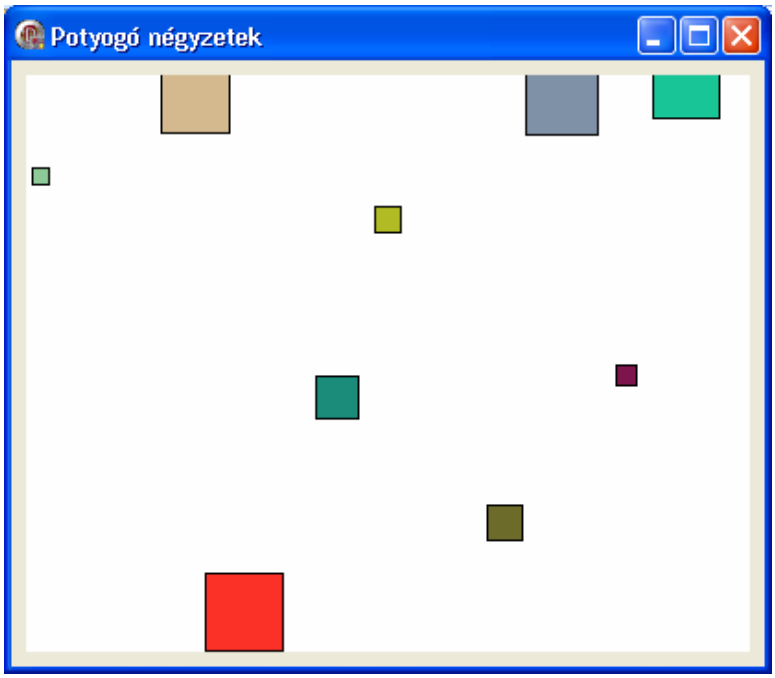


4. Definiáljon külön unitban egy **TNegyzet** osztályt, melynek legyenek következő tulajdonságai: x, y koordinátája, oldalának hossza, kitöltés színe (természetesen más tulajdonsága is lehet, ha szükséges). Tartalmazzon egy **Mozgat** metódust, melynek meghívásakor a négyzet egy pixellel

lejjebb "pottyan" a képernyőn. Írjuk át az osztályhoz tartozó **konstruktort** is, melynek paramétereként adjuk megy hogy melyik Image komponensre szeretnénk kirajzolni a négyzeteket. A konstruktor generáljon ki véletlen számokat az x, y koordinátákra, az oldalhosszra és a kitöltés színére. Szükség szerint további metódusokat is tartalmazhat.

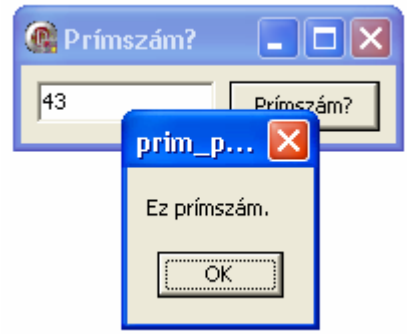
A program indításakor ebből az osztályból **készítsen 10 objektumot**, melyeket helyezzen el egy Image komponensen. Az alkalmazás bezárásakor ne felejtse el felszabadítani az objektumoknak lefoglalt memóriát.

Az objektumok **Mozgat** metódusainak folyamatos meghívásával a négyzetek "potyogjanak" a képernyőn amíg a program fut. Ha valamelyik négyzet teljesen elhagyja a képernyő alját, akkor a képernyő tetejéről "jöjjön be" más x koordinátával, más mérettel és más színnel (véletlenszerű).



5. Készítsünk egy dinamikus csatolású könyvtárat (DLL-t), amely tartalmaz egy „prímszám” függvényt. Ez a függvény a paraméterében megadott egész számról döntse el, hogy az prímszám-e és ettől függően adjon vissza igaz vagy hamis értéket.

   A megírt DLL-t felhasználva készítsük el az alábbi alkalmazást, amely egy nyomógomb megnyomásakor megvizsgálja, hogy az Edit komponensben megadott szám prímszám-e.



6. Készítsünk alkalmazást prímszámok generálására, amely egy nyomógombot, egy ListBox-ot és egy Gauge komponest tartalmaz. A nyomógomb megnyomása után a program a ListBox-ba generálja ki az első 20000 prímszámot. A Gauge komponens folyamatosan jelezze, hogy a 20000 prímszámnak eddig hány százaléka található a ListBox-ban. A nyomógomb megnyomása után a prímszámok generálását (majd beírását a ListBox-ba és a Gauge komponens frissítését) egy külön programszálban végezzük el! Próbáljuk úgy megírni az algoritmust, hogy az a számok generálását minél hamarabb elvégezze.