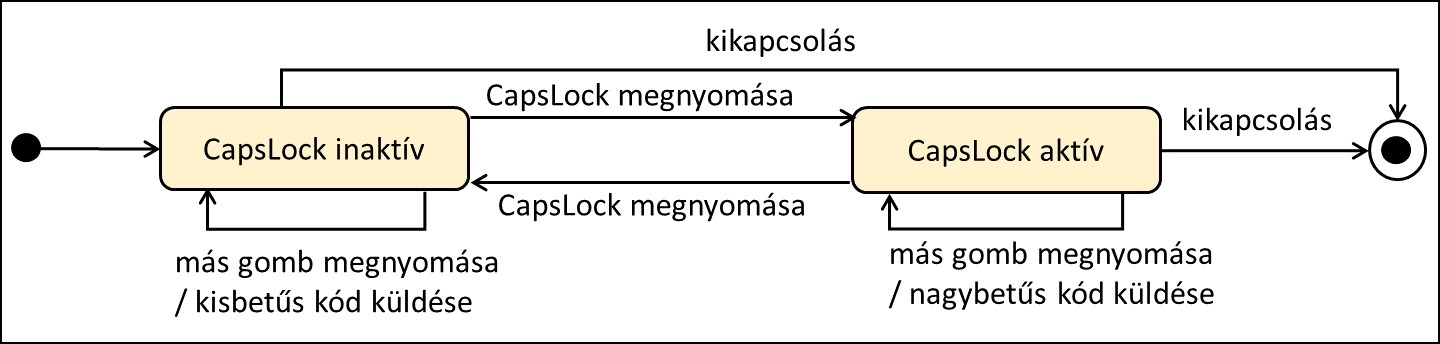
***Állapotgépek***

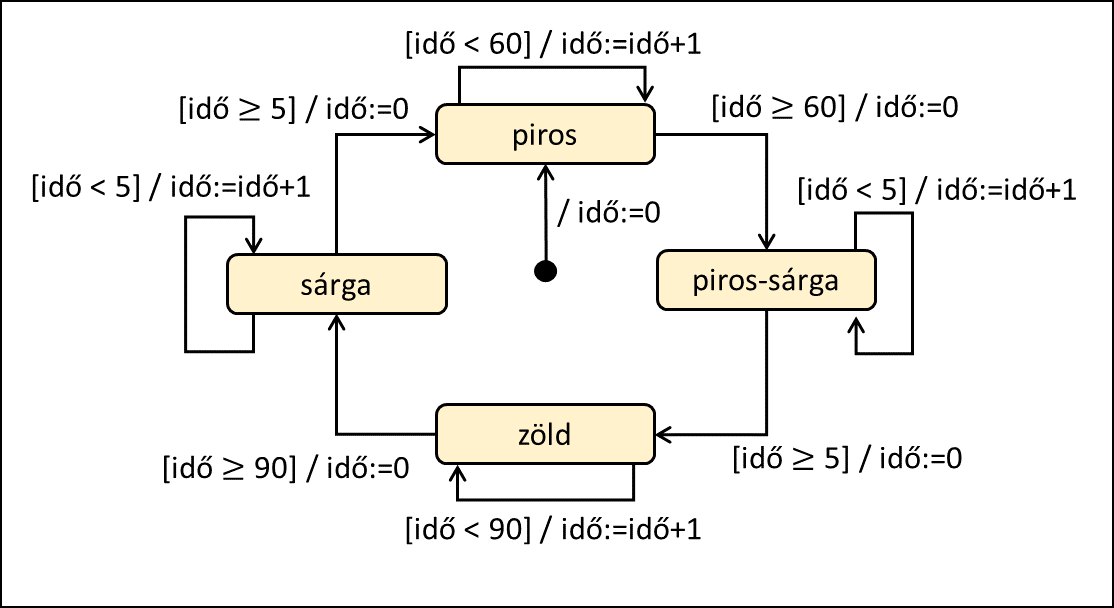
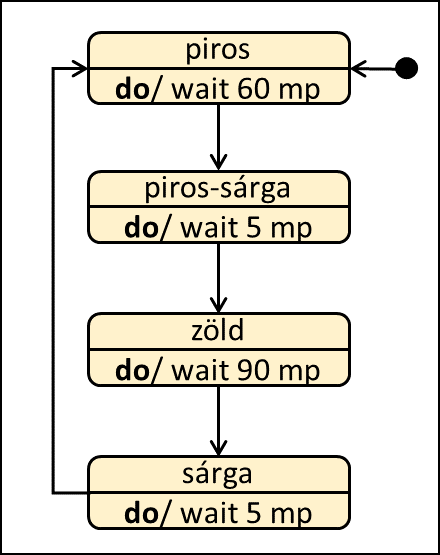
Egy állapotgép felrajzolásához mindig társul egy osztálydiagramm, hiszen az objektumoknak van állapotgépe, és az objektumokat osztálydiagram segítségével jellemezzük.

1. Egyszerűsített billentyűzet modellezése: ha a CapsLock aktivált, akkor minden más billentyű lenyomásra nagybetűs karaktereket, különben kisbetűs karaktereket kapunk. (Legyen két állapot: CapsLock aktív, illetve inaktív. Legyen háromféle művelet: CapsLock lenyomása, más gomb lenyomása, kikapcsolás.)

Legyen két állapot: CapsLock aktív, illetve inaktív. Legyen háromféle művelet: CapsLock lenyomása, más gomb lenyomása, kikapcsolás



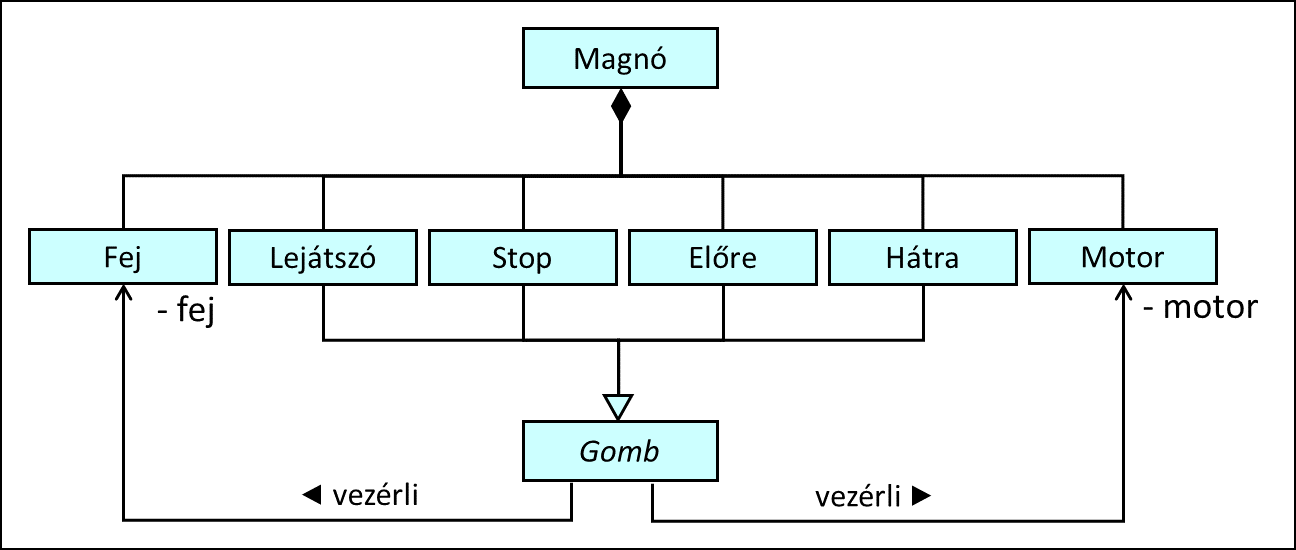
1. Egy közlekedési lámpán piros, piros-sárga, zöld, sárga fények vannak. A lámpa 60 másodpercig piros és 90 másodpercig zöld színű. Az átmeneti állapotok 5 másodpercig tartanak: pirosról a zöldre a piros-sárgán keresztül, zöldről a pirosra a sárgán keresztül. Kezdetben a lámpa piros.

1. Készítsük el egy videomagnó osztálydiagramját és állapotgépét! A magnóban található egy olvasó fej és egy motor, amelyeket négy gomb segítségével vezérelhetünk. A gombokat elegendő megérinteni a vezérlés során. A négy gomb és vezérlési szerepük a következő:

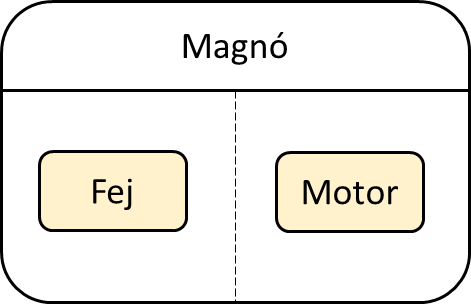
* ⏹(állj) : leállítja a motort, és a fejet felemeli a szalagról, ha azon volt.;
* ⏵(lejátszás) : lejátszó sebességbe helyezi a motort, és a fejet a szalagra helyezi, ha még nincs ott.
* ⏩(előre) : a motor előre csévéli a szalagot
* ⏪(hátra) : a motor hátra csévéli a szalagot
* Előre, illetve hátracsévélés alatt a fej rajta maradhat a szalagon: ez a gyorskeresés funkció.

Osztálydiagram:

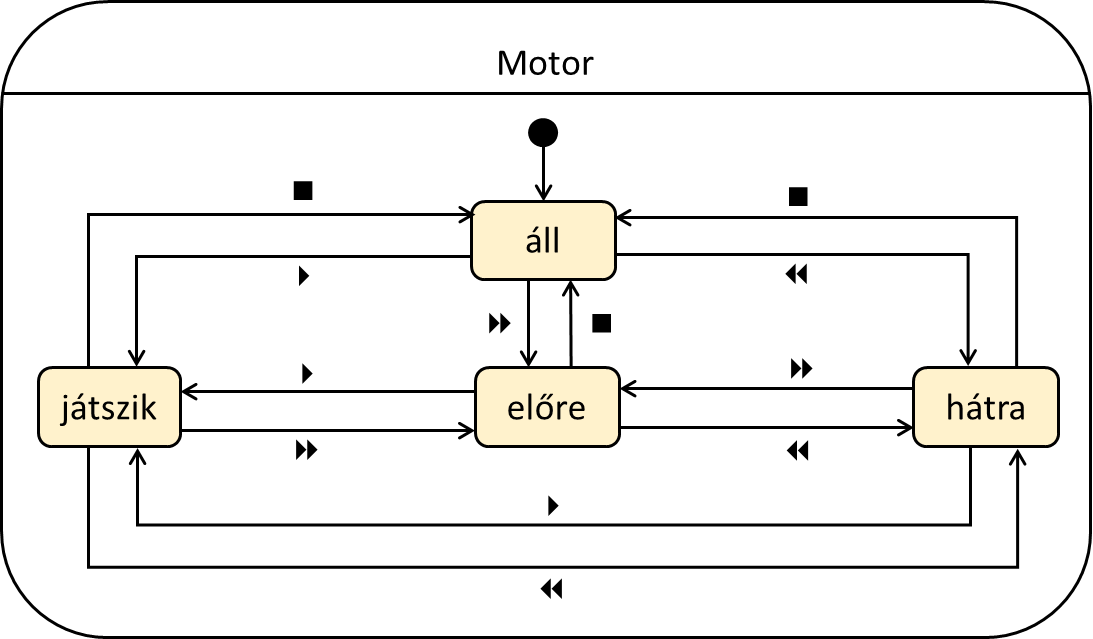
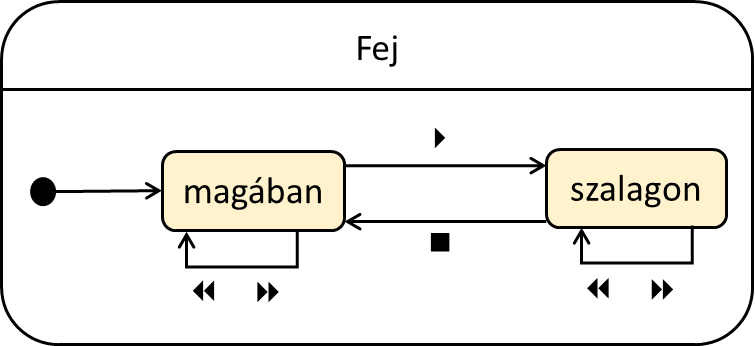


A gombok a fejnek és a motornak küldenek (⏹, ⏵, ⏩, ⏪) szignálokat. Ha a „vezérli” asszociációkban bevezetjük a privát fej és motor szerepneveket, akkor a szignálküldést a **send** ⏹**to** fej, **send**⏹**to** motor, **send**⏵**to** fej, stb. utasításokkal végezhetjük.

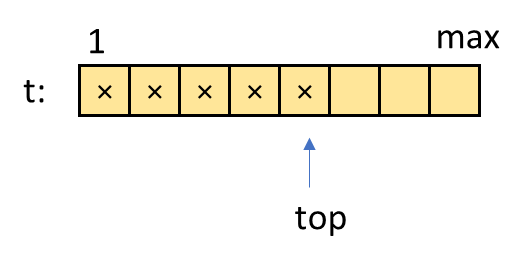
Összetett állapotgép:



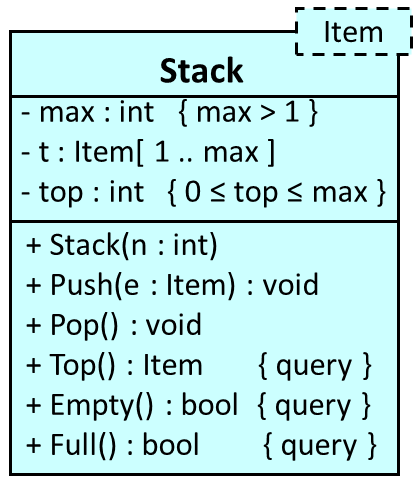
Komponens állapotgépek:

1. Készítsünk egy tárolót, amely veremként (LIFO) működik a Push() és Pop() műveletekkel. Az előre nem meghatározott típusú elemeket egy rögzített méretű (max) tömb (t : Item1..max) tárolja, amelynek az első és a top-adik közötti részén helyezkednek el a veremben tárolt elemek: az első a verem alja, a top-adik a teteje.



Osztálydiagram:



Állapotgép:

Három állapotot vezetünk be: „üres” (top=0), „köztes” (0<top<max), „tele” (top=max), amelyek között a verem műveletek (Push(), Pop()) hatására következik be átmenet. Kezdetben (kezdeti átmenet) top:=0.

A képen szöveg, képernyőkép, Betűtípus, diagram látható

Automatikusan generált leírás

Az állapotgép tervéből kiolvasható a verem konstruktorának, valamint a Push() és Pop() metódusainak működése.

1. Készítsünk egy sorként (FIFO) működő tárolót előre nem meghatározott típusú (Item) elemek számára az Enqueue() és Dequeue() műveletekkel. Az elemeket a sorba kerülésük sorrendjében egy tömbben (t : Item0..max-1) tároljuk úgy, hogy a sor elejét (a legrégebben betett elem indexét) egy out adattag, a sor végét (a legutoljára betett elem utáni hely indexét) egy in adattag tárolja. A sor elemeinek számát (count) külön is nyilvántartjuk.

A képen képernyőkép, Téglalap, sor látható

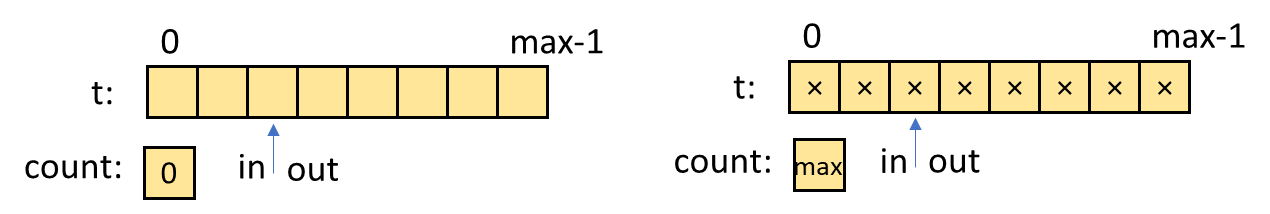
Automatikusan generált leírás

Ha az out<in, akkor a sor elemei az out-adik helytől kezdődően az in-edik előtti helyig helyezkednek el: az out indexű helyen a legrégebben sorban levő elem található (ez a sor eleje); az in indexű helyre kell majd betenni a következő elemet (ez a sor vége). Az Enqueue() művelet az in, a Dequeue() művelet az out értékét növeli eggyel.

A képen képernyőkép, Téglalap, sor látható

Automatikusan generált leírás

Érdemes a sor elejét és végét mutató indexeket ciklikusan (+1 mod max) növelni az Enqueue(), illetve a Dequeue() műveleteknek: így előállhat olyan eset is, hogy az out>in. Ekkor a sorban levő elemek egyrészt a tömb out-adik helyétől a tömb végéig, másrészt a tömb elejétől az in-edik előtti helyig helyezkednek el.



Különleges eset az, amikor in=out, mert ilyenkor nem lehet eldönteni, hogy egy üres sor reprezentációját, vagy max darab elemet tartalmazó sor reprezentációjával van-e dolgunk. Ezért kiegészítjük a reprezentációt egy count adattaggal, amelyik a sorban levő elemek számát mutatja.

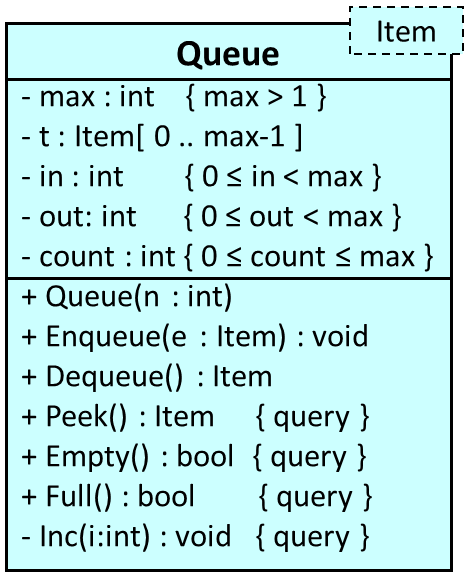
Fontos látni azt a kapcsolatot (típus invariánst), amely az adattagok között áll fenn:

in-out ha in>out

count = max-(out-in) ha in<out

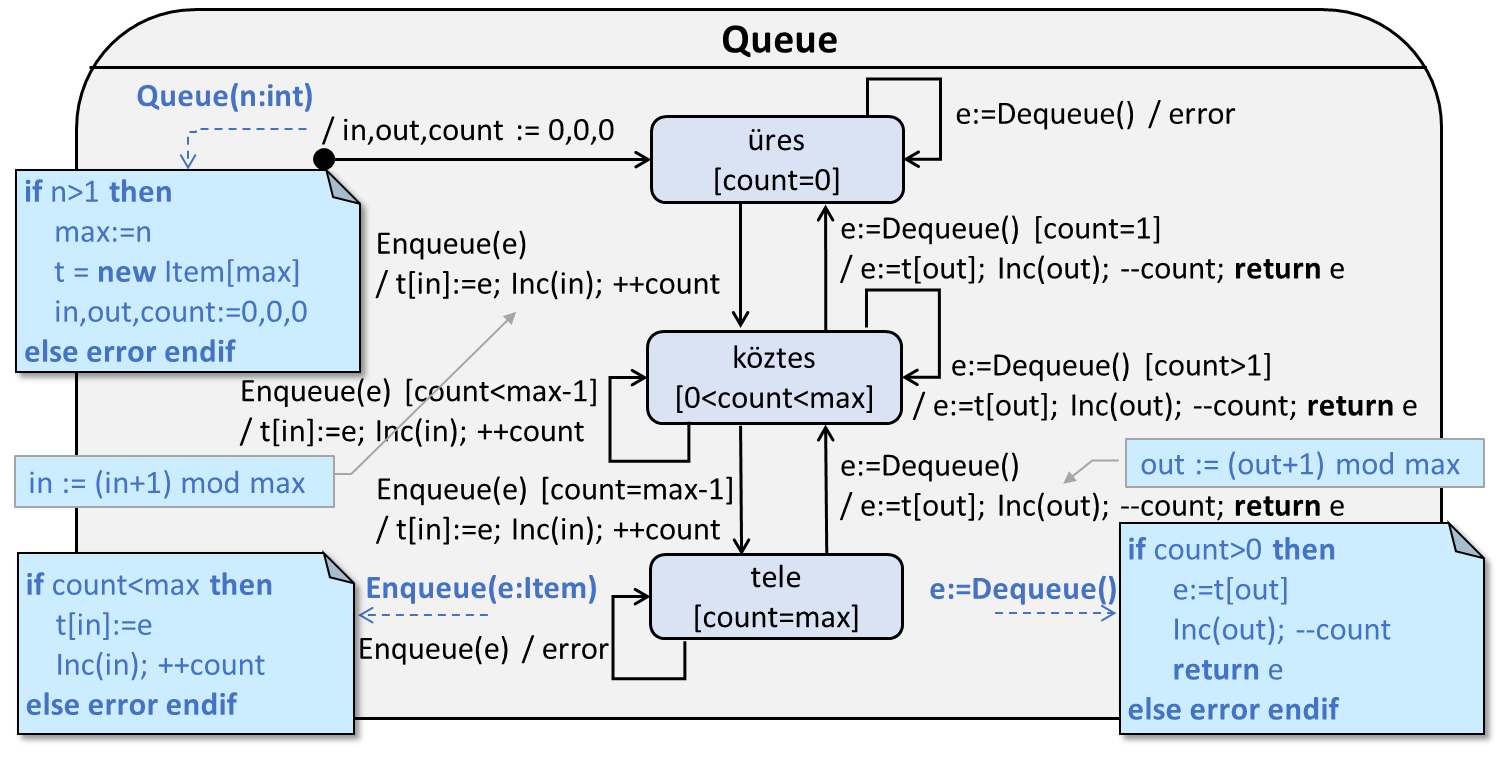
max vagy 0 ha in=out

Osztálydiagram:

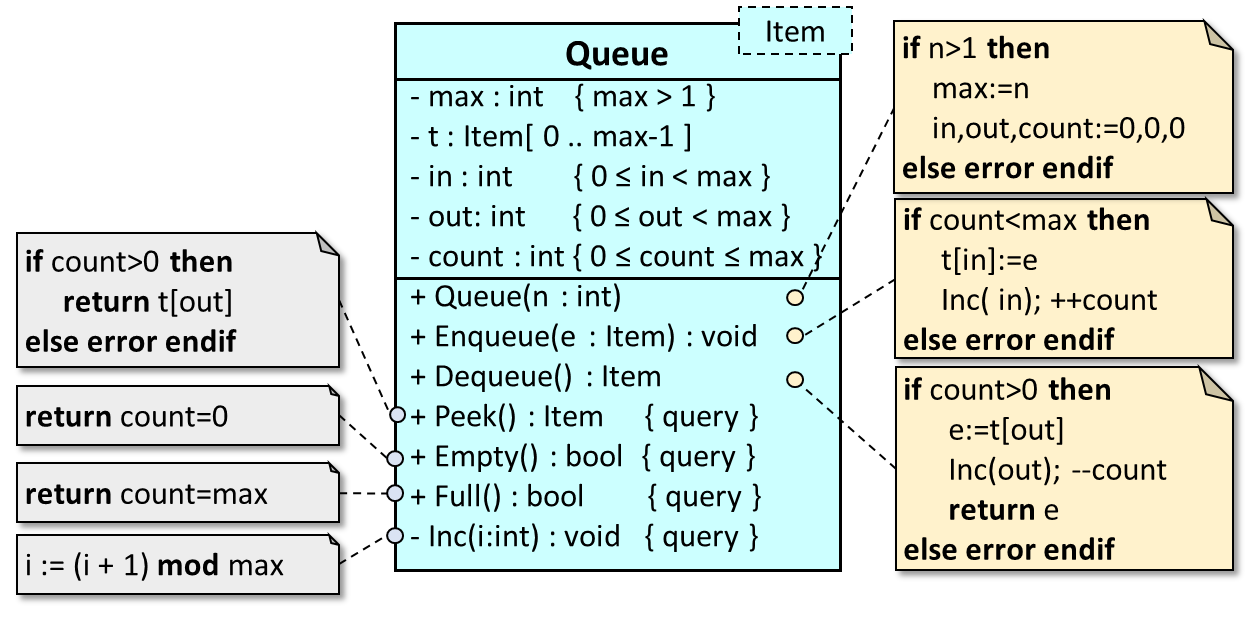


Állapotgép:

Három állapotot vezetünk be: „üres” (count=0), „köztes” (0<count<max, amihez fel kell tennünk, hogy a max>1), „tele” (count=max), amelyek között a sor műveletek (Enqueue(), Dequeue()) hatására következik be átmenet. Kezdetben (kezdeti átmenet) cout:=0, amely mellé az invariáns miatt in=out is kell, például in:=0 és out:=0.



Az állapotgép tervéből kiolvasható a sor konstruktorának, valamint az Enqueue() és a Dequeue() metódusainak működése.



1. Egy mikrohullámú sütő meghatározó elemei az ajtó, a lámpa, egy vezérlő gomb és a magnetron. A vezérlő gomb megnyomásával indítjuk el a magnetront, feltéve, hogy az ajtó csukva van, és ilyenkor a lámpa is világítani kezd. A magnetron működését vagy a vezérlő gomb megnyomásával tudjuk leállítani, ilyenkor a lámpa is kialszik, vagy az ajtó kinyitásával, de ilyenkor a lámpa égve marad. Az ajtó kinyitása mindig felkapcsolja a lámpát, bezárása pedig lekapcsolja.

A használati eset diagram a rendszer fő tevékenységeit mutatja meg.

A képen szöveg, képernyőkép, diagram, Betűtípus látható

Automatikusan generált leírás

Szekvencia diagram egyfelől helyettesíti az objektum- és kommunikációs diagramot, másfelől egy lehetséges forgatókönyvet mutat be a mikrohullámú sütő működtetésére. A korábbi feladatokhoz képest újdonság, hogy az objektumok között kommunikációban jelentős szerep jut a szignáloknak, amely a közvetlen metódushívásoknak az alternatívája. Nem törvényszerű, de követendő, hogy egy objektumhoz vagy csak szinkron üzenet (metódushívás formájában), vagy csak aszinkron üzenet (szignál formájában) érkezzen.

A képen szöveg, képernyőkép, szám, diagram látható

Automatikusan generált leírás

Az osztálydiagram asszociációi mentén valósul meg a kommunikáció. (Külön jelöltük azokat, ahol szignál-küldésre kerül sor.) A kommunikációt segítő szerepneveket, azok láthatóságát, gondosan kell megtervezni.

A képen szöveg, képernyőkép, diagram, képernyő látható

Automatikusan generált leírás

A rendszer működését egy összetett állapotgéppel jellemezzük.

A képen szöveg, képernyőkép, képernyő, Betűtípus látható

Automatikusan generált leírás