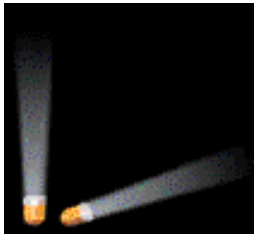
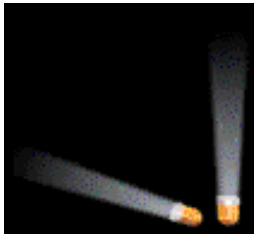
Дууриамал Заагч





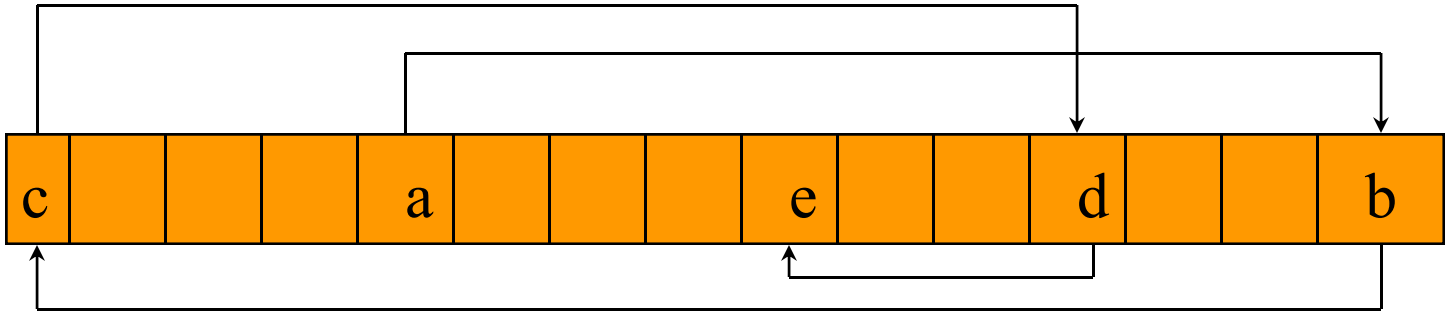
Java -ийн Заагчийн дутагдал

- Зөвхөн өгөгдлийн дотоод бүтцэт ашиглагддаг.
- Өгөгдлийн бүтцийг сэргээхэд дараалалжуулалт(serialization) хэрэгтэй болдог.
- Арифметик биш

Жишээ

- Дискийн халгалах байгууламж – FAT (File Allocation Table)
- Интернетээр өгөгдөл дамжуулах – TCP/IP Protocol

Дууриамал-Заагч Ойн байршил



Өгөгдлийн бүтэц ойд массив хэлбэртэй, түүний байршил бүрт element (Object төрөл) болон next (int төрөл) гэсэн талбаруудтай.

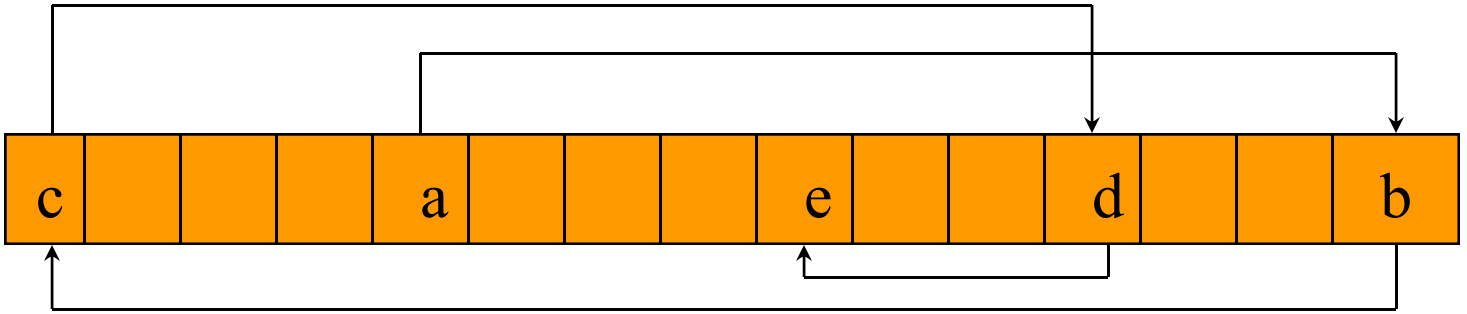
Зангилааны дүрслэл

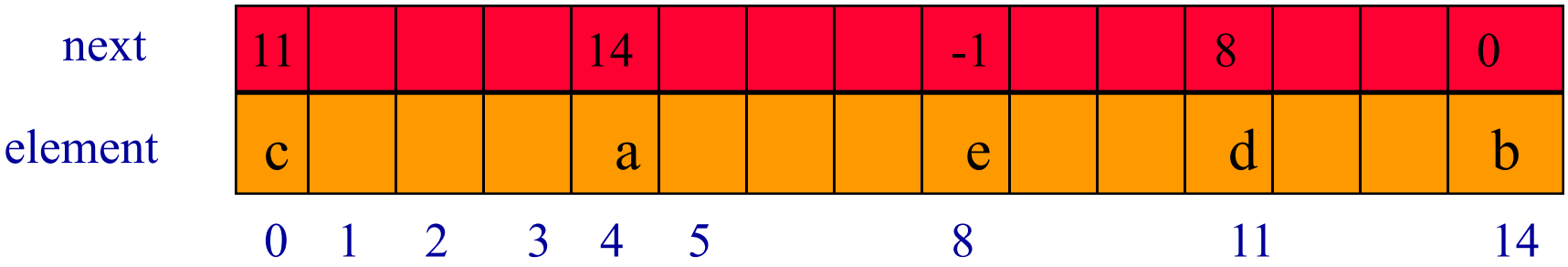
```
package dataStructures;


class SimulatedNode
{
   // өгөгдөл гишүүд
   Object element;
   int next; //Анхаарлаа энд хандуулна уу


    // байгуулагч энд бичигдэнэ
}
```

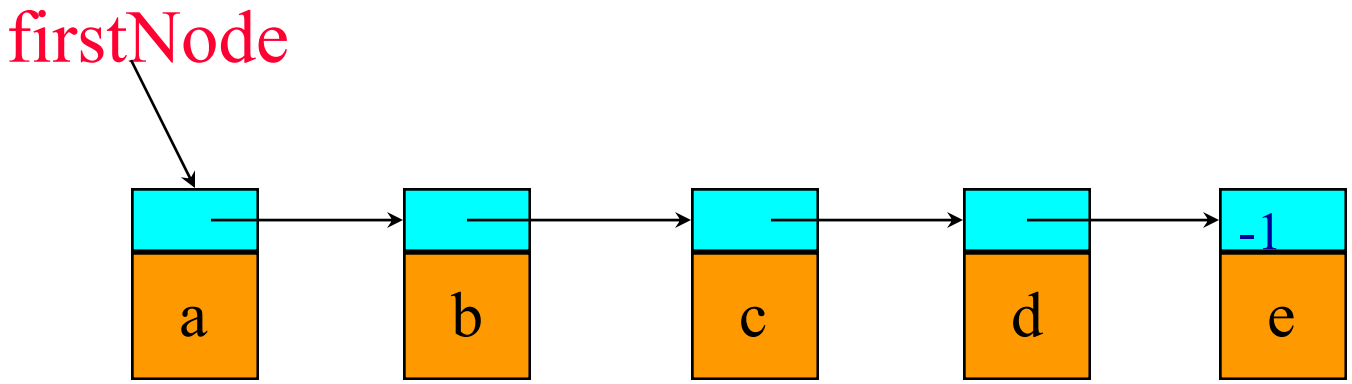
Энэ бүхэн яаж харагдах вэ





firstNode = 4

Зурагдахдаа холбоост дүрслэлтэй адил байна



Ойн удирдлага

Холбоос (Java эсхүл дууриамал заагч) дараах зүйлийг шийдэхийг шаардана:

- ашиглагдаагүй ойг хянах (ө.х, санах байгууламжийн нөөц)
- зангилаа хуваарилах

  Java -д new гэсэн арга бий
- ашиглагдахгүй болсон зангилааг чөлөөлөх

  Java хаягдал цуглуулах механизмтай

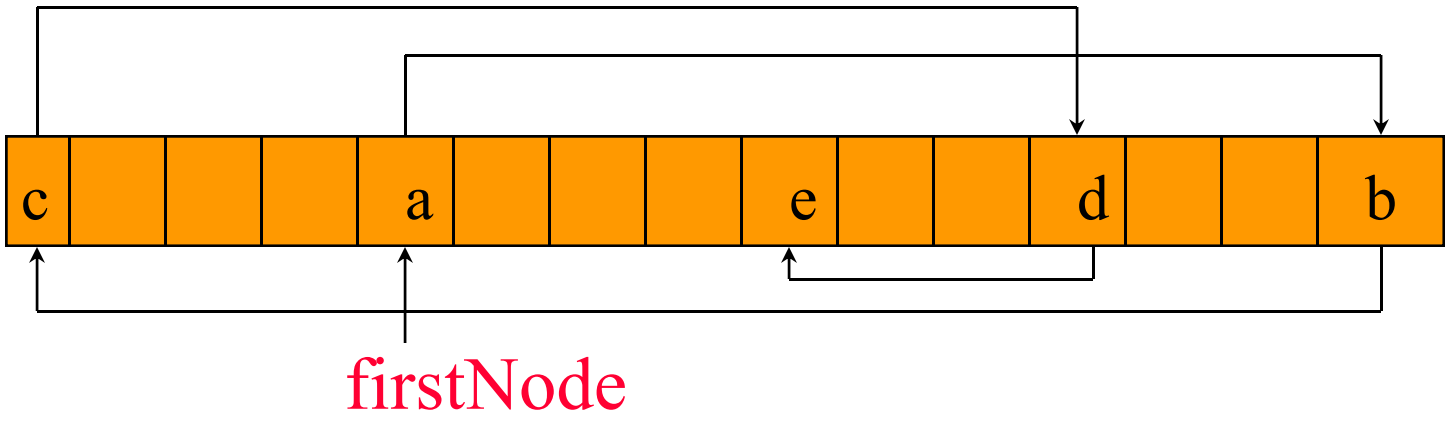
Хаягдал цуглуулах

Систем ашиглагдаагүй зангилаа/ойг шүүрдэж санах байгууламжийн чөлөөт нөөцөд буцаадаг. (сүүдрийн горимонд ажиллах !)

Үүнийг хоёр/гурван алхмаар хийдэг:

- Ашиглаагүй зангилааг тэмдэглэх.
- Сул зайг нягтруулах (заавал биш). – дискийн дефрагментацтай төстэй
- Чөлөөлөгдсөн занилааг санах байгууламжийн нөөцөд шилжүүлэх.

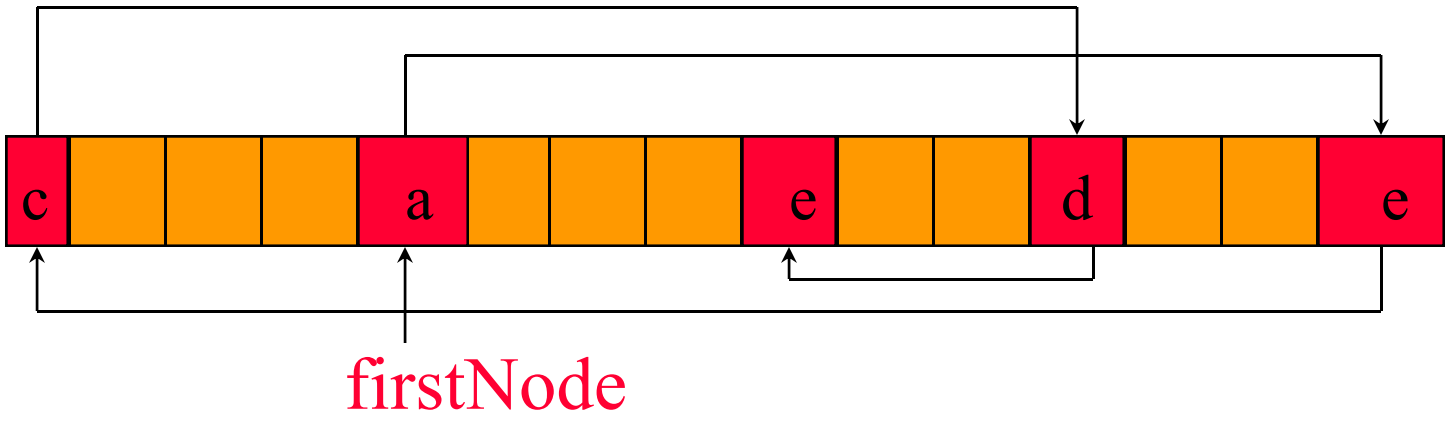
Тэмдэглэх



Бүх зангилааг тэмдэггүй болго (бүх тэмдэгийн битүүдийг false болгох).

Хаяг агуулсан хувьсагч бүрээс эхлэх бүх заагчийг даган очсон зангилаа бүрээ тэмдэглэнэ.

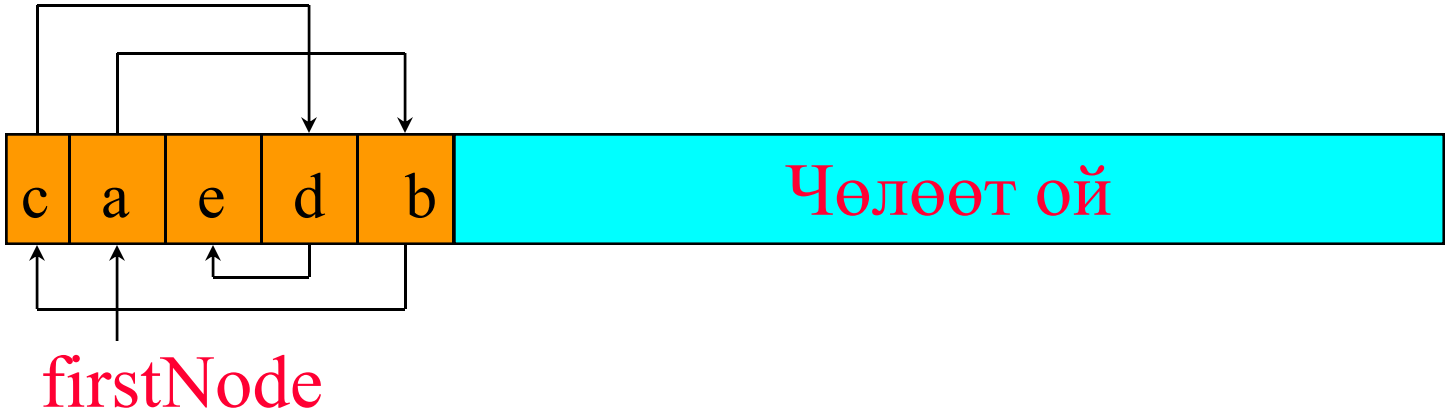
Тэмдэглэх



firstNode –ээс эхлэн firstNode –оос хүрж болох бүх зангилааг тэмдэглэнэ.

Үүнийг хаягийн бүх хувьсагчийн хувьд давтана.

Нягтруулах



Бүх тэмдэглэгдсэн зангилааг (ө.х., ашиглагдаж байгаа зангилаанууд) ойн аль нэг төгсгөл рүү шилжүүлэх, шаардлагатай заагч нарыг өөрчлөх.

Суларсан ойг санах байгууламжийн нөөцөд хийх

Тэмдэглэгдээгүй зангилааг ойд шүүнэ (хэрвээ нягтруулалт хийгдээгүй бол), эсхүл нэгэн чөлөөт блокыг (сул ой байгаа бол) нөөцөөд хийнэ.

Хаягдал цуглуулахын давуу тал

- Програм зохиогч суларсан зангилааг чөлөөлөхөд санаа тавих хэрэггүй.
- Гэхдээ хэрэггүй болсон объектыг илэрхийлж байгаа хувьсагчийг null болгохоос нааш хаягдал цуглуулалт ажиллахгүй.

Хаягдал цуглуулахын давуу тал

- Ой ихтэй компьютер дээр програм хурдан ажиллах боломжтой.

Хаягдал цуглуулахын сул тал

- Хаягдал цуглуулах хугацаа ойн хэмжээтэй шууд хамааралтай (сул ойн хэмжээтэй хамаагүй).

Хаягдал цуглуулах алтернатив арга

Зангилааг чөлөөлөх/суллах аргатай болох.

өх., C++ -ийн delete арга

Ингэснээр чөлөөлөгдсөн зангилаа үргэлж санах байгууламжийн нөөцөд орох болно.

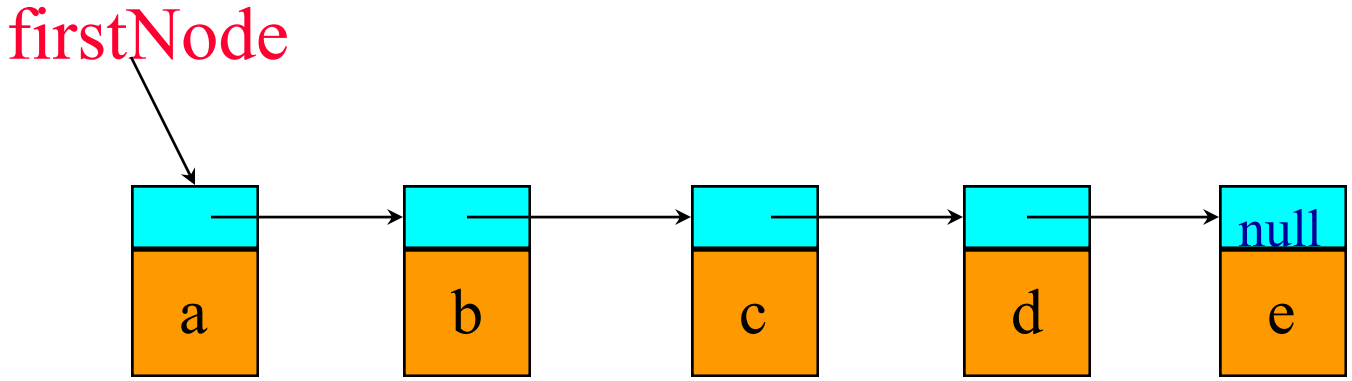
Алтернатив аргын давуу тал

- Зангилааг чөлөөлөх хугацаа нийт ойн хэмжээ бус, чөлөөлөгдсөн зангилааны тоотой пропорционал байдаг.

Алтернатив аргын сул тал

- Хэрэглэгч өгөгдлийн бүтцийн зангилааг чөлөөлөх аргыг бичих хэрэгтэй.
- Зангилааг чөлөөлөхөд зарсан хугацаа дахин ашиглагдахгүй.
- Ойн хэмжээг нэмэгдүүлэх нь програмын ажиллах хугацааг богиносгодрггүй.

Зангилаа бүхэн ижил хэмжээтэй бол санах байгууламжийн нөөцийн зохион байгуулалт



- Чөлөөлөгдсөн зангилааны гинжийг зхицуулах
- Гинжний эхнээс хуваарилах
- Чөллөгдсөн зангилааг гинжний эхэнд нэмэх

Дууриамал Загчийн Ойн Удирдлага

```
/** дууриамал заагчийн ойг удирдах класс */
package dataStructures;
import utilities.*;
public class SimulatedSpace1
{
   // өгөгдөл гишүүд
   private int firstNode;
   SimulatedNode [] node;  // багцад харагдана

   // байгуулагч болон бусад аргууд
}
```

Байгуулагч

```
public SimulatedSpace1(int numberOfNodes)
 {
   node = new SimulatedNode [numberOfNodes];

   // зангилааг үүсгэж гинжинд холбох
   for (int i = 0; i < numberOfNodes - 1; i++)
     node[i] = new SimulatedNode(i + 1);

   // массив болон гинжийн сүүлийн зангилаа
   node[numberOfNodes - 1] = new SimulatedNode(-1);
   // firstNode –ийн анхны утга 0
 }
```

Зангилаа Хуваарилах

```
public int allocateNode(Object element, int next)
  {// Хоосон зангилаа хуваарилж, талбаруудын утгыг тогтоох.
    if (firstNode == -1)
    {   // энд бичигдэх кодыг орхилоо
    }

    int i = firstNode;              // анхны зангилааг хуваарилах
    firstNode = node[i].next;  // firstNode дараагийн чөлөөт
   зангилааг заана
    node[i].element = element;
    node[i].next = next;
    return i;
  }
```

Зангилааг Чөлөөлөх

```
public void deallocateNode(int i)
  {// Зангилаа i –г чөлөөлөх.
    // i -г хоосон жагсаалтын эхний зангилаа болгох
    node[i].next = firstNode;
    firstNode = i;

    // элементийн хаягаар илэрхийлэгдэх орон зайг хаягдалд өгч
    //  чөлөөлөх
    node[i].element = null;
  }
```

Дууриамал Заагч

- Дахин холбоос үүсгэхгүйгээр зангилаануудын гинжийг хуваарилж болно.
- Гинжний эхлэл болон төгсгөлийн зангилаа мэдэгдэж байвал зангилааны гинжийг богино хугацаанд чөлөөлж болно.

Дууриамал Заагч

- Java –ийн хаягаас дууриамал заагч илт давуу талтай болох нь харагдахгүй байвал ашиглах хэрэггүй.