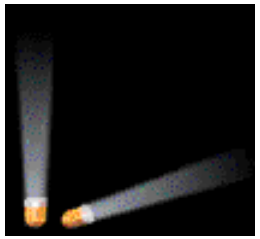
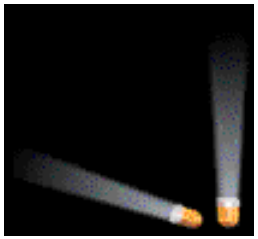
F.CS203 - Өгөгдлийн бүтэц ба алгоритм 2022-2023

Д.Золбоо

МТ-н салбарын багш

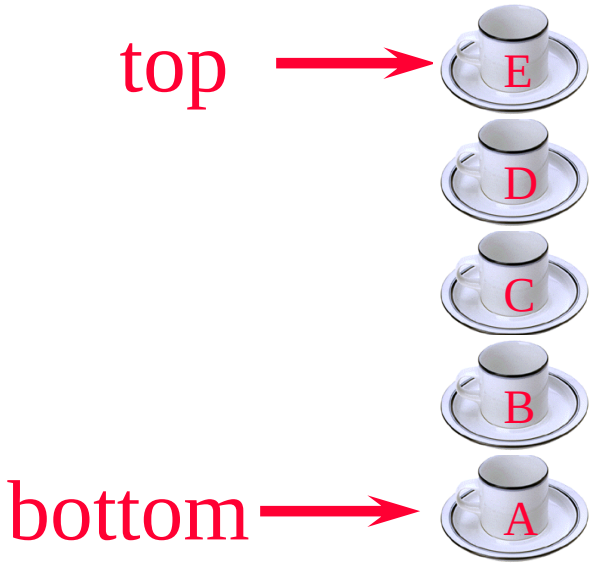
Стек

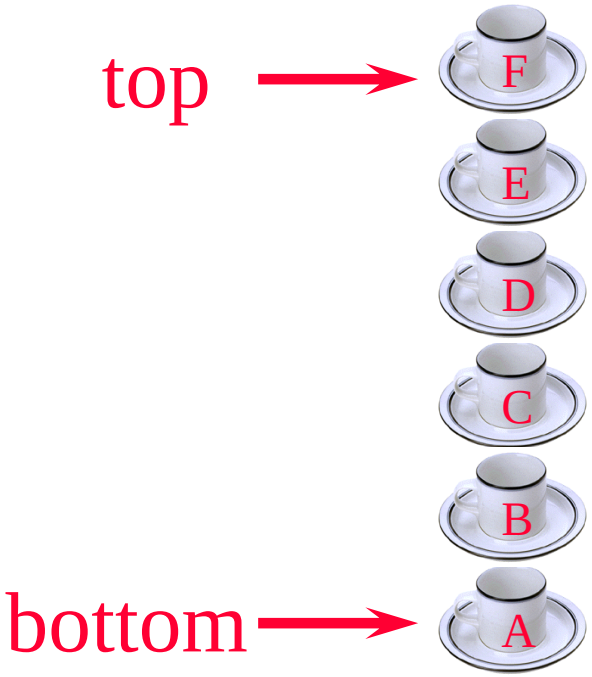




- Шугаман жагсаалт.
- Нэг төгсгөлийг нь top-орой.
- Нөгөө төгсгөлийг нь bottom-ёроол.
- Нэмэлт ба хасалтыг зөвхөн оройгоос нь хийнэ.

Аяганы стек





- Стект аяга нэмэх.
- Стекээс аяга авах.
- Стек бол LIFO жагсаалт.

Интерфейс - Stack

```
public interface Stack
{
    public boolean empty();
    public Object peek();
    public void push(Object theObject);
    public Object pop();
}
```

Хаалт хослох

- (((a+b)*c+d-e)/(f+g)-(h+j)*(k-l))/(m-n)
  - u байршлын нээх хаалтад харгалзах v байршлын хаах хаалтын хослолыг (u,v) гэж бичвэл
    - (2,6) (1,13) (15,19) (21,25) (27,31) (0,32) (34,38)
- (a+b))*((c+d)
  - (0,4)
  - байршил 5 –ын хаах халтад харгалзах нээх хаалт алга
  - (8,12)
  - байршил 7 –ийн нээх хаалтад харгалзах хаах хаалт алга

Хаалт хослох

- Илэрхийллийг зүүнээс баруун тийш шинжих
- Нээх хаалт тааралдвал түүний байршлыг стект нэмнэ
- Хаах хаалт тааралдвал харгалзах байршлыг стекээс авна

Жишээ

- (((a+b)*c+d-e)/(f+g)-(h+j)*(k-l))/(m-n)

Жишээ

- (((a+b)*c+d-e)/(f+g)-(h+j)*(k-l))/(m-n)

(2,6) (1,13)

Жишээ

- (((a+b)*c+d-e)/(f+g)-(h+j)*(k-l))/(m-n)

(2,6) (1,13) (15,19)

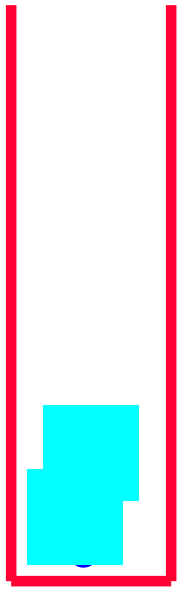
Жишээ

- (((a+b)*c+d-e)/(f+g)-(h+j)*(k-l))/(m-n)

(2,6) (1,13) (15,19) (21,25)

Жишээ

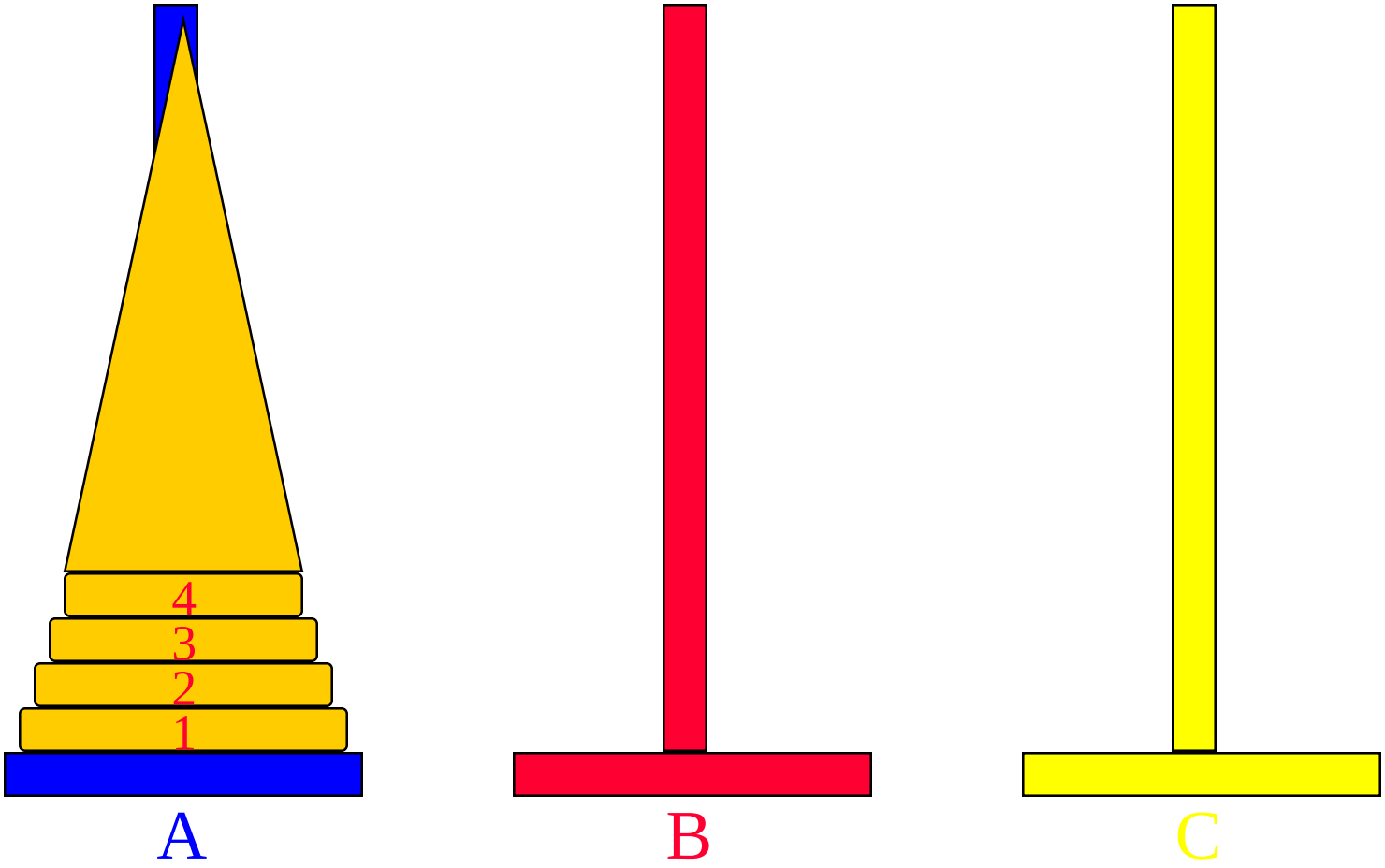
- (((a+b)*c+d-e)/(f+g)-(h+j)*(k-l))/(m-n)



(2,6) (1,13) (15,19) (21,25)(27,31) (0,32)

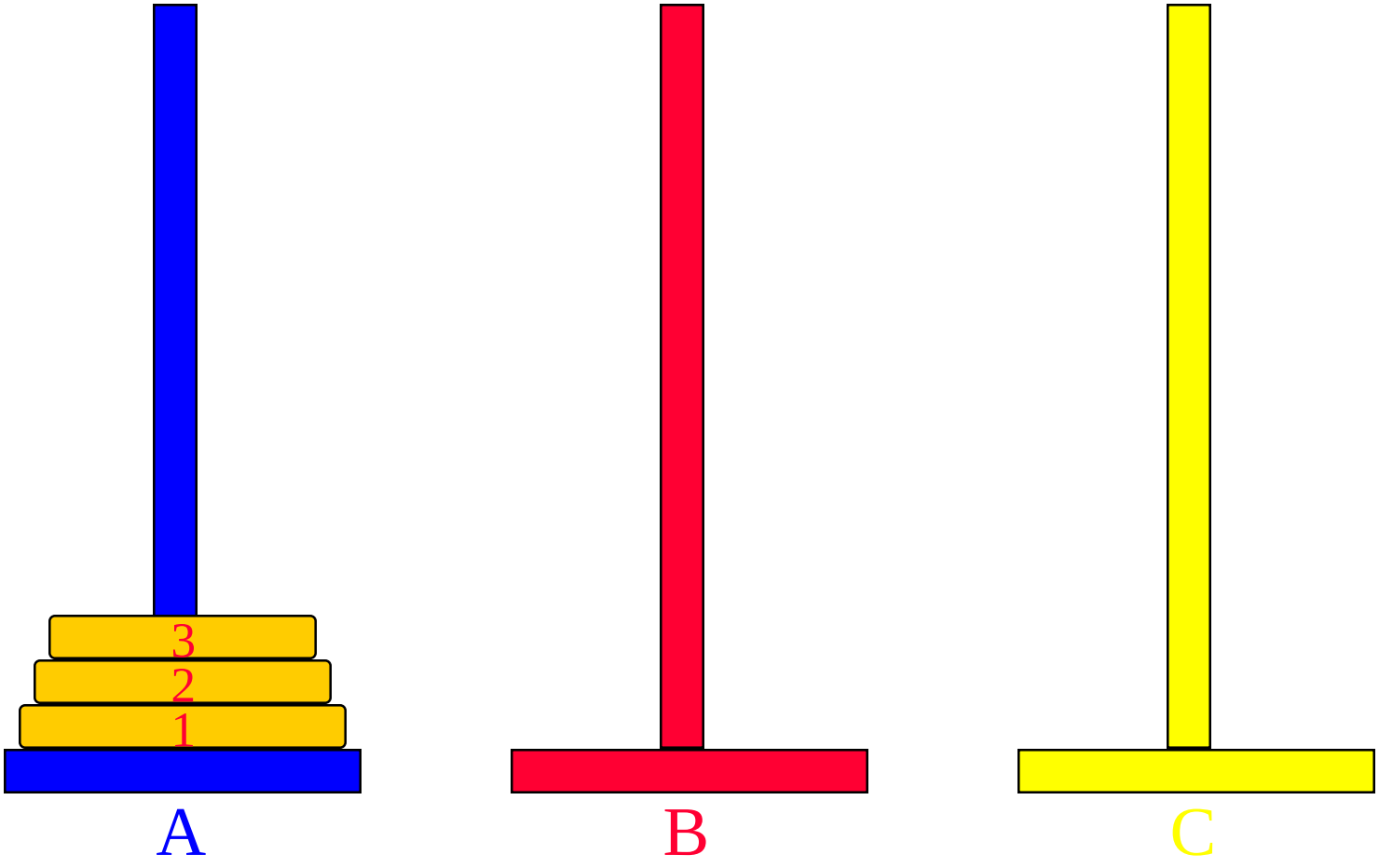
- ГЭХ МЭТ

Ханойн цамхаг



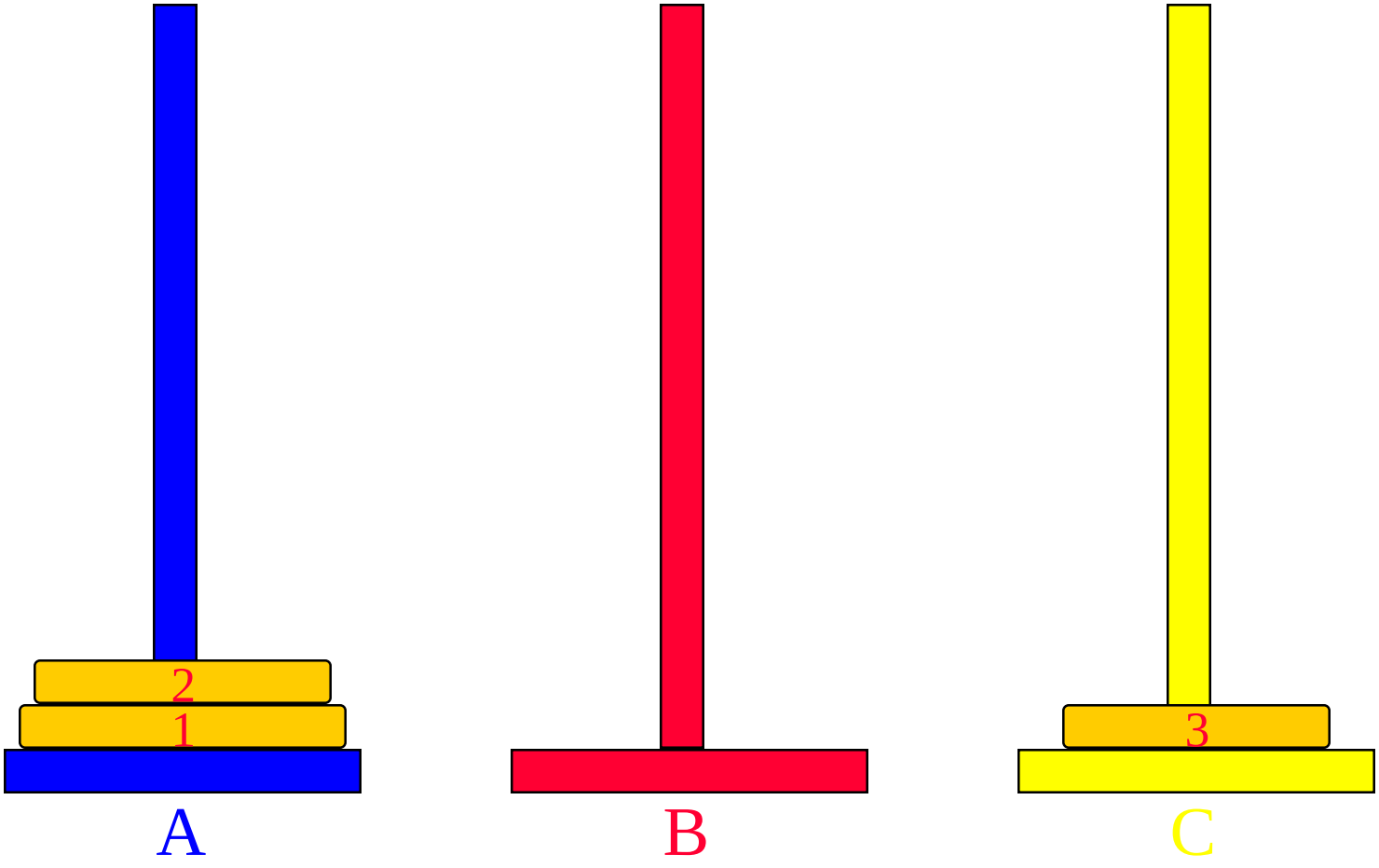
- 64 алтан дискийг A цамхгаас C цамхагт шилжүүлнэ
- Цамхаг бүр стек шиг ажиллана
- Том дискийг жижиг диск дээр тавьж болохгүй

Ханойн цамхаг



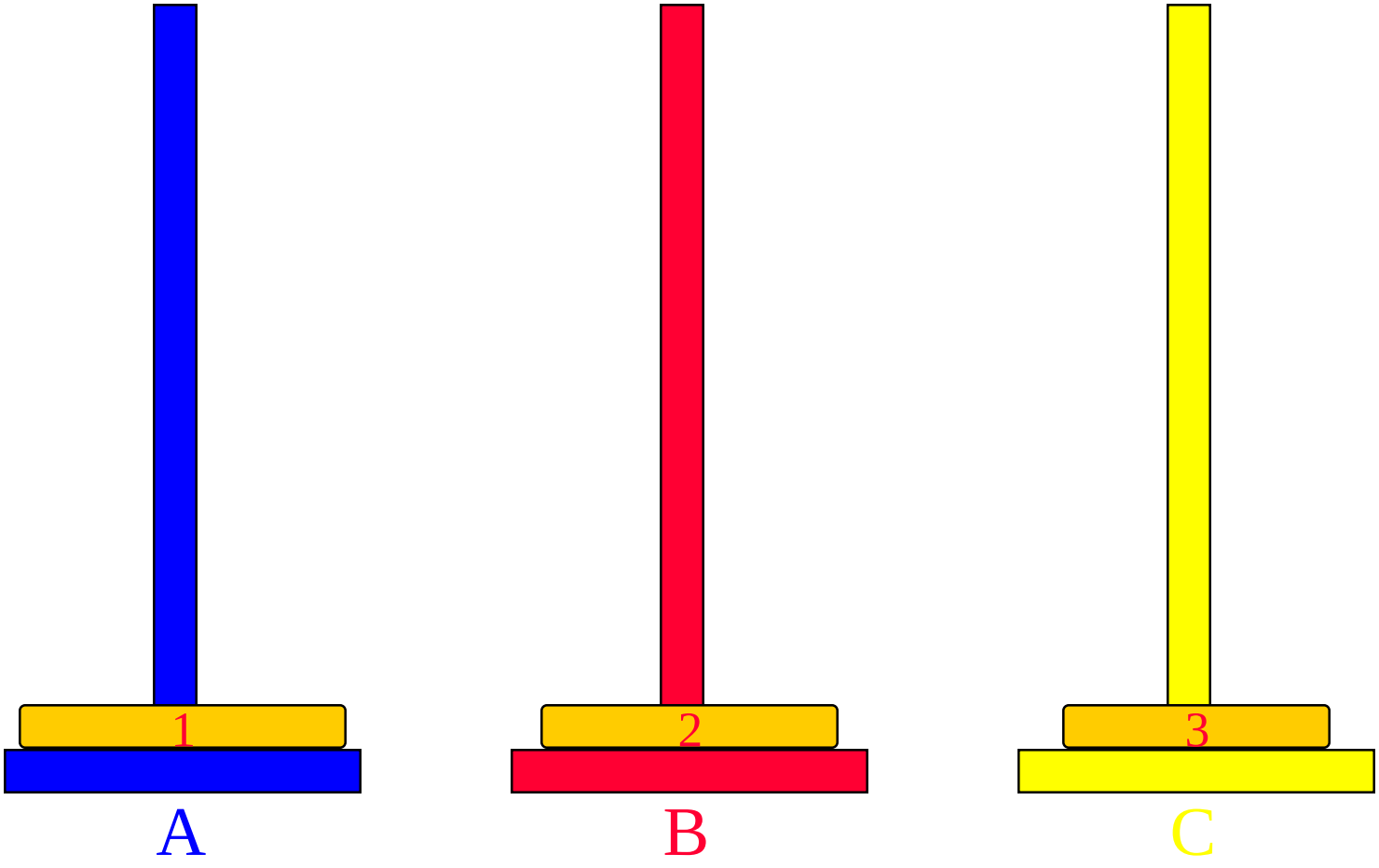
- 3 дисктэй Ханойн цамхаг

Ханойн цахаг



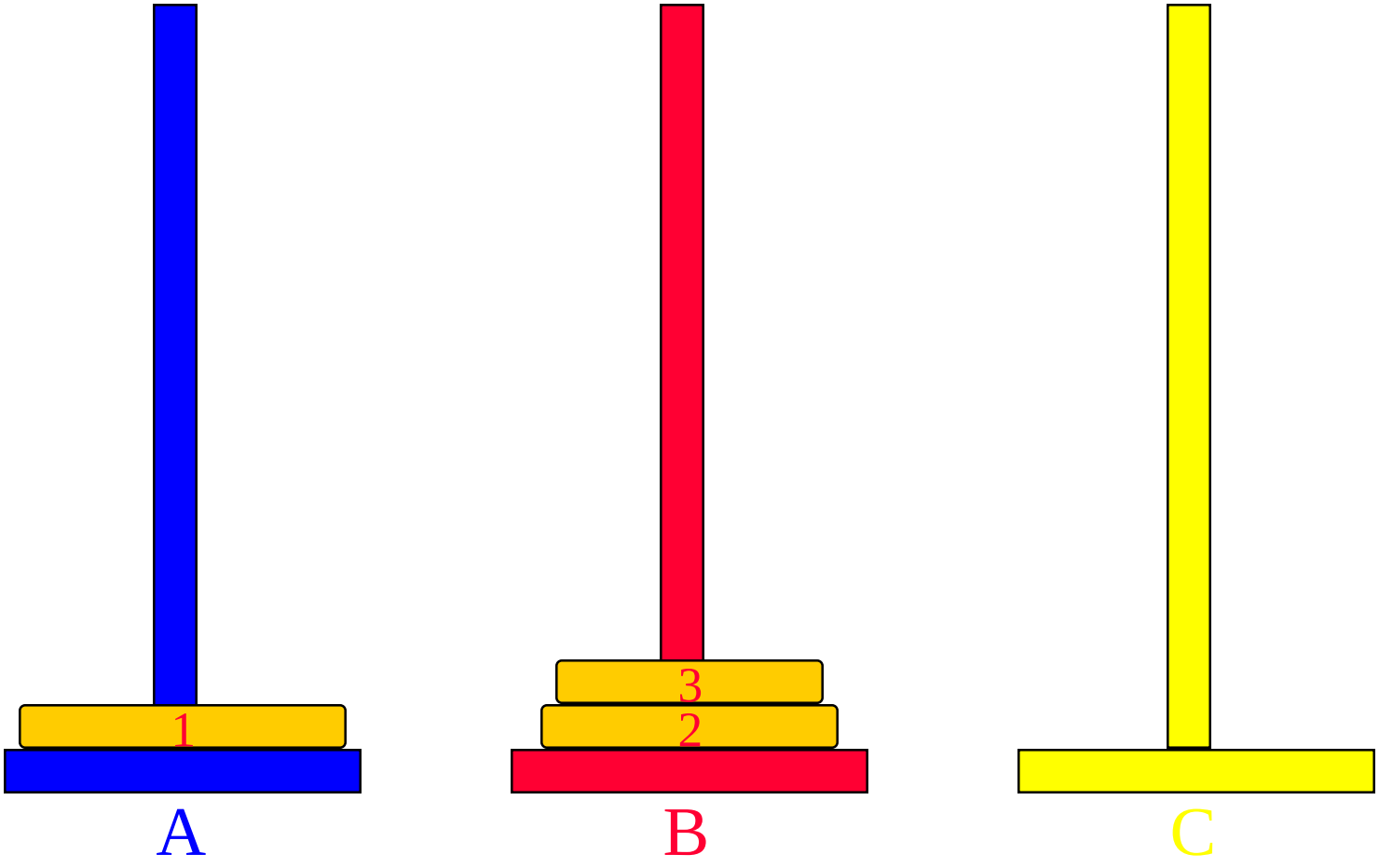
- 3 дисктэй Ханойн цамхаг

Ханойн цамхаг



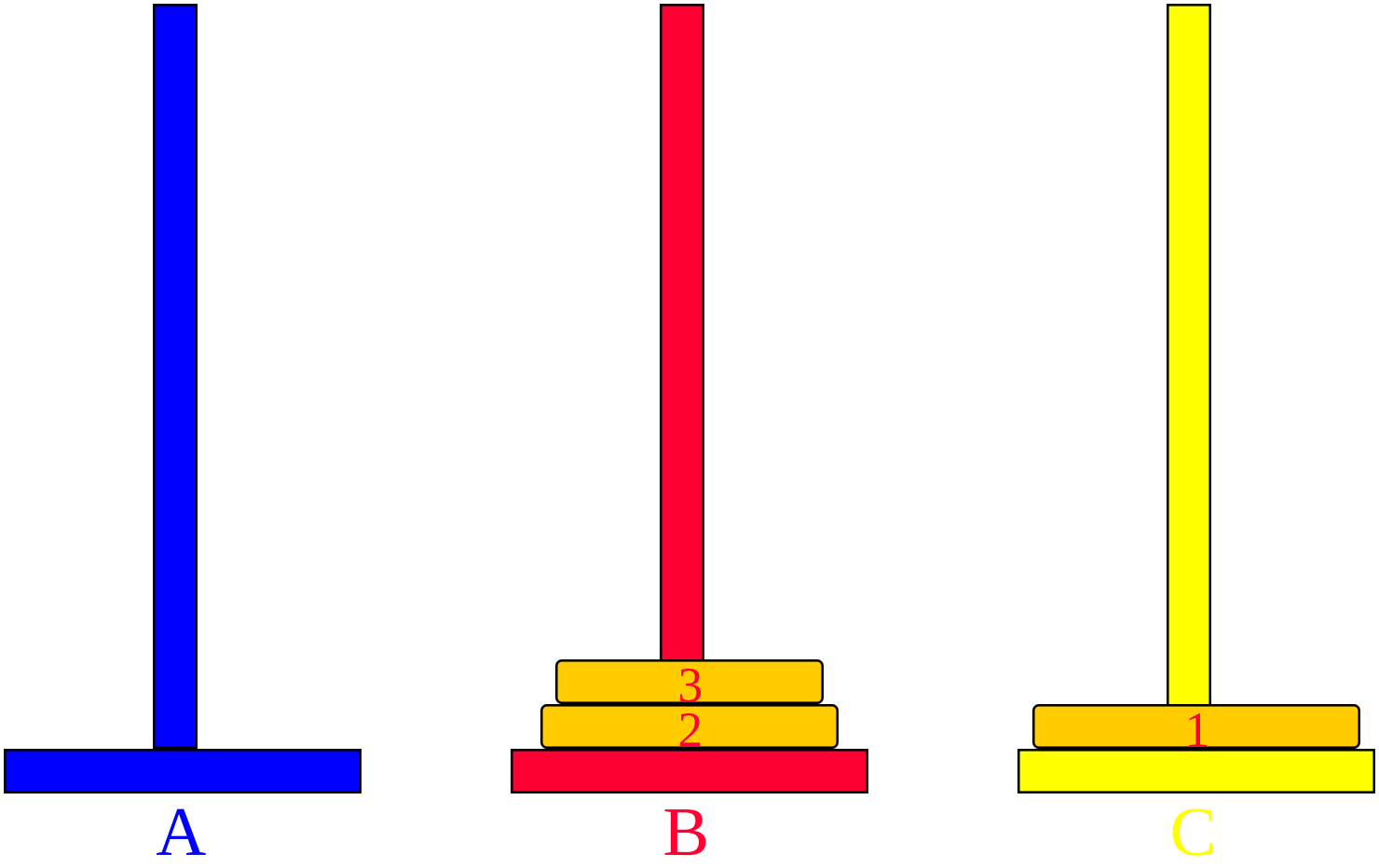
- 3 дисктэй Ханойн цамхаг

Ханойн цамхаг



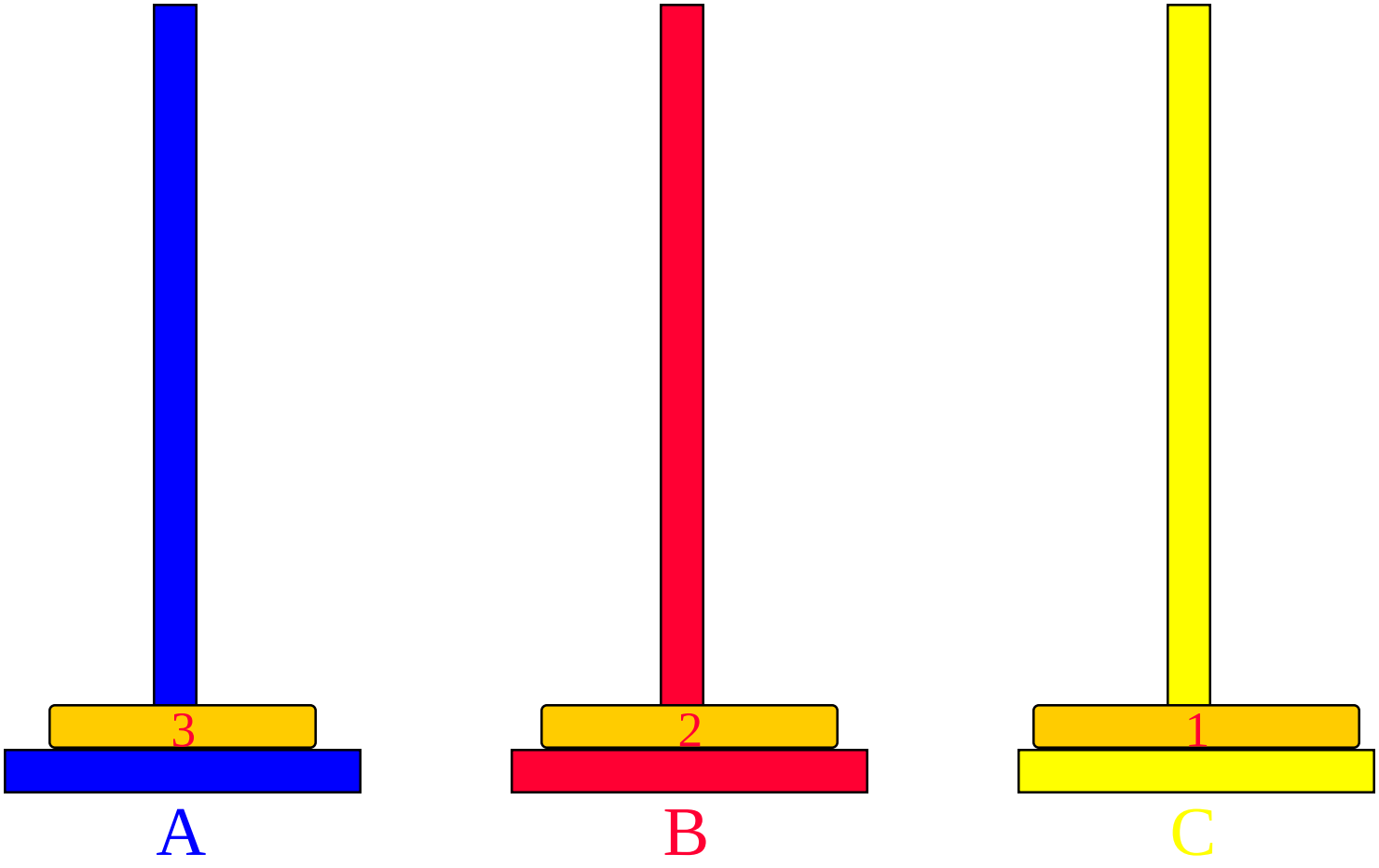
- 3 дисктэй Ханойн цамхаг

Ханойн цамхаг



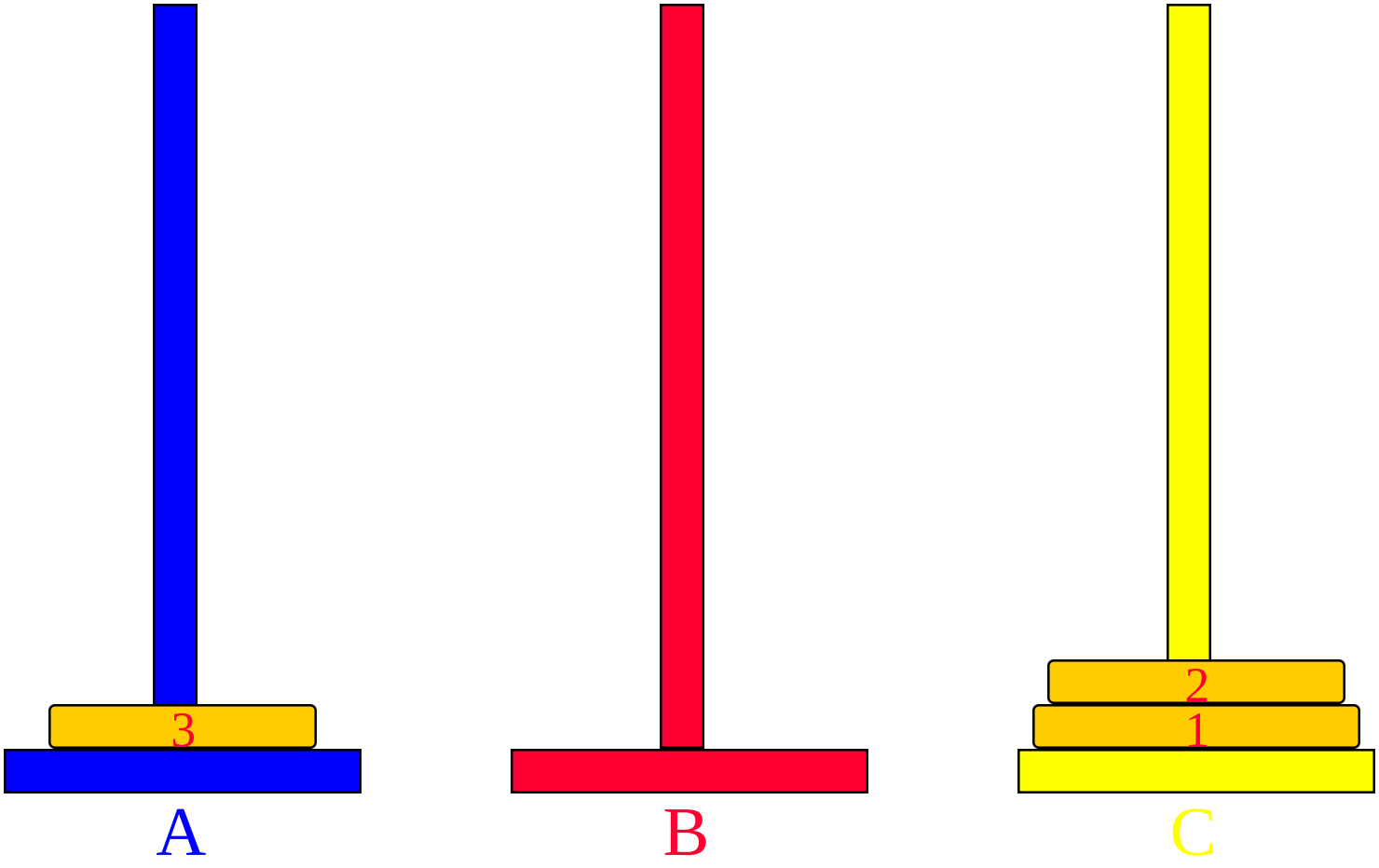
- 3 дисктэй Ханойн цамхаг

Ханойн цамхаг



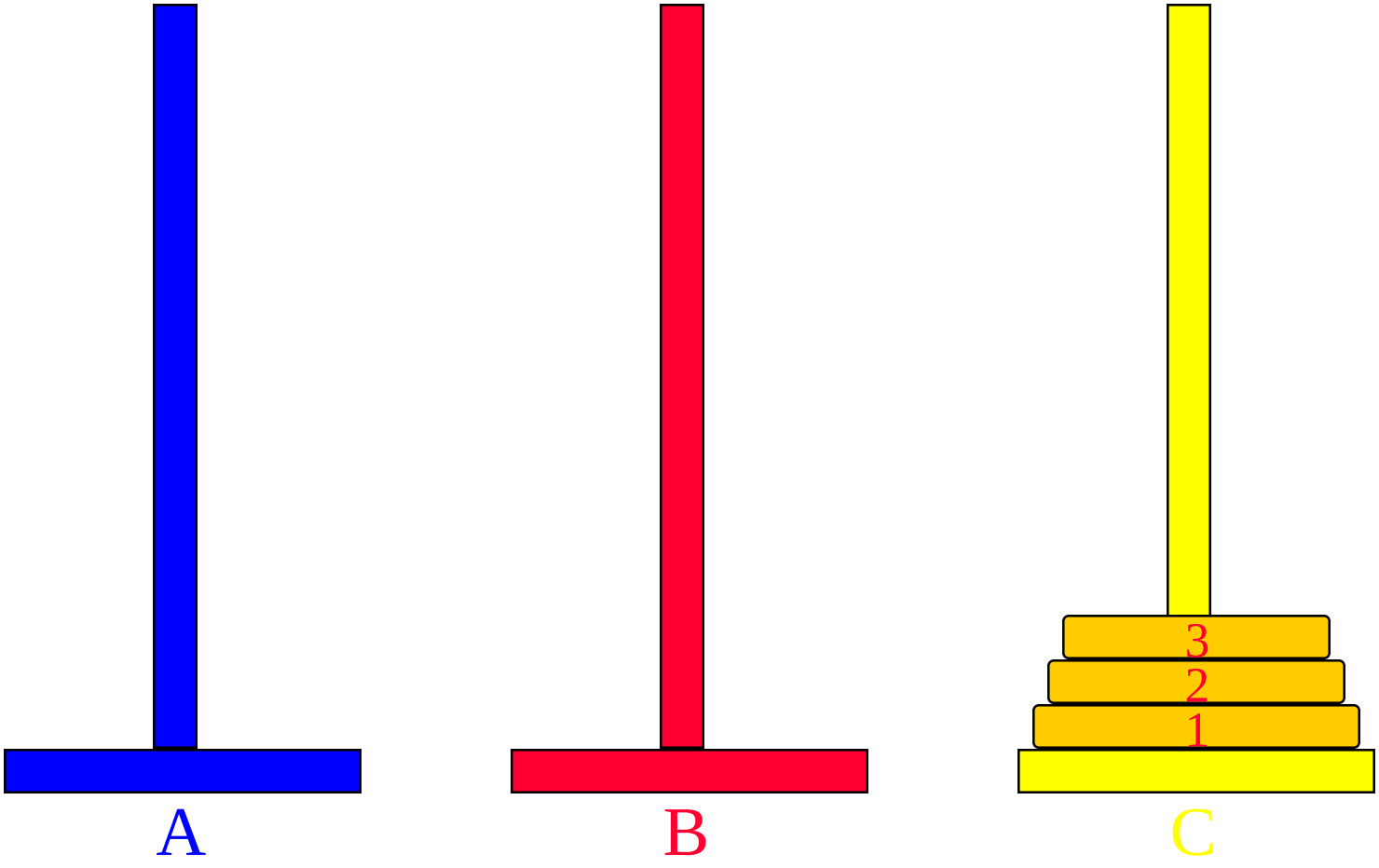
- 3 дисктэй Ханойн цамхаг

Ханойн цамхаг



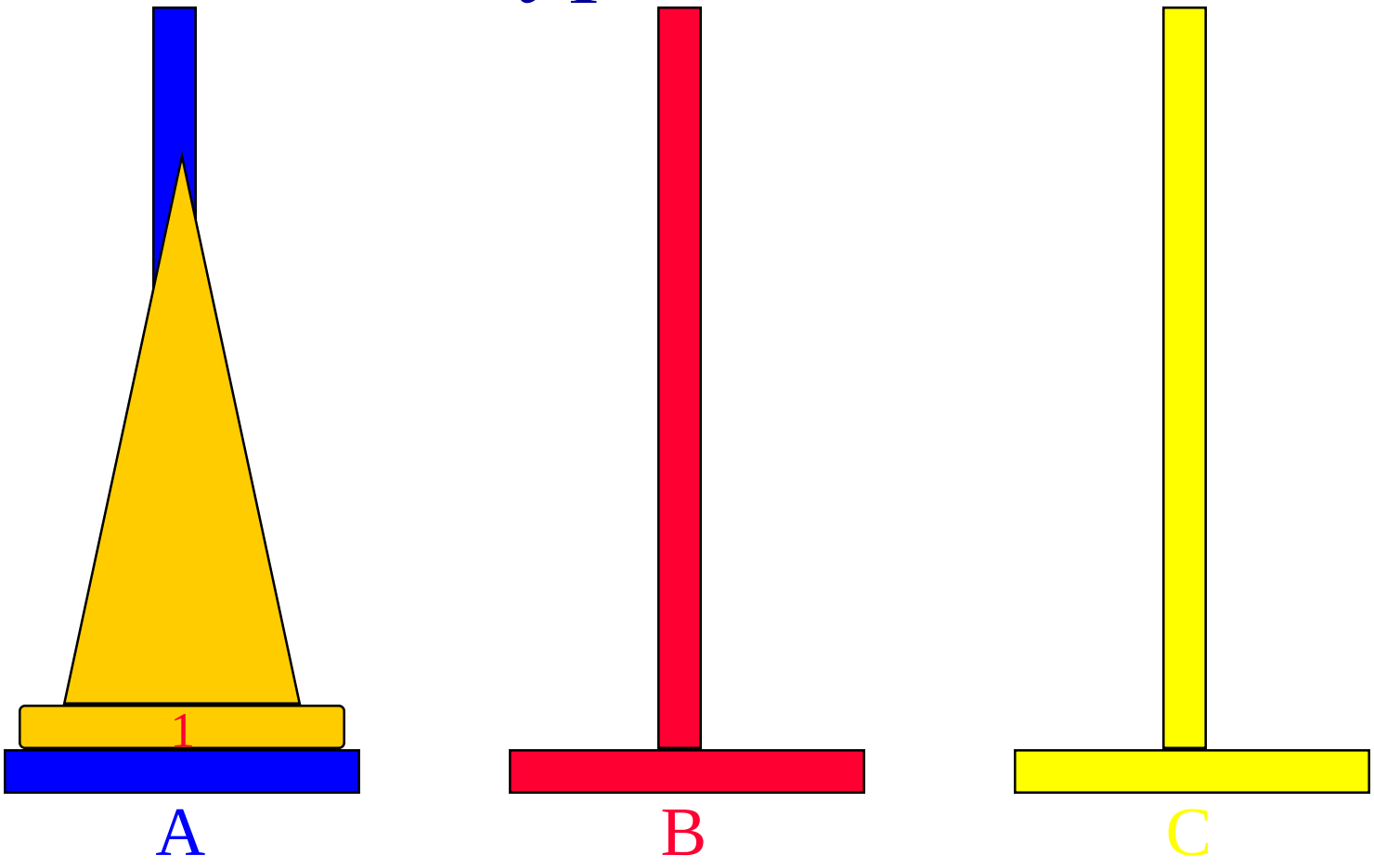
- 3 дисктэй Ханойн цамхаг

Ханойн цамхаг



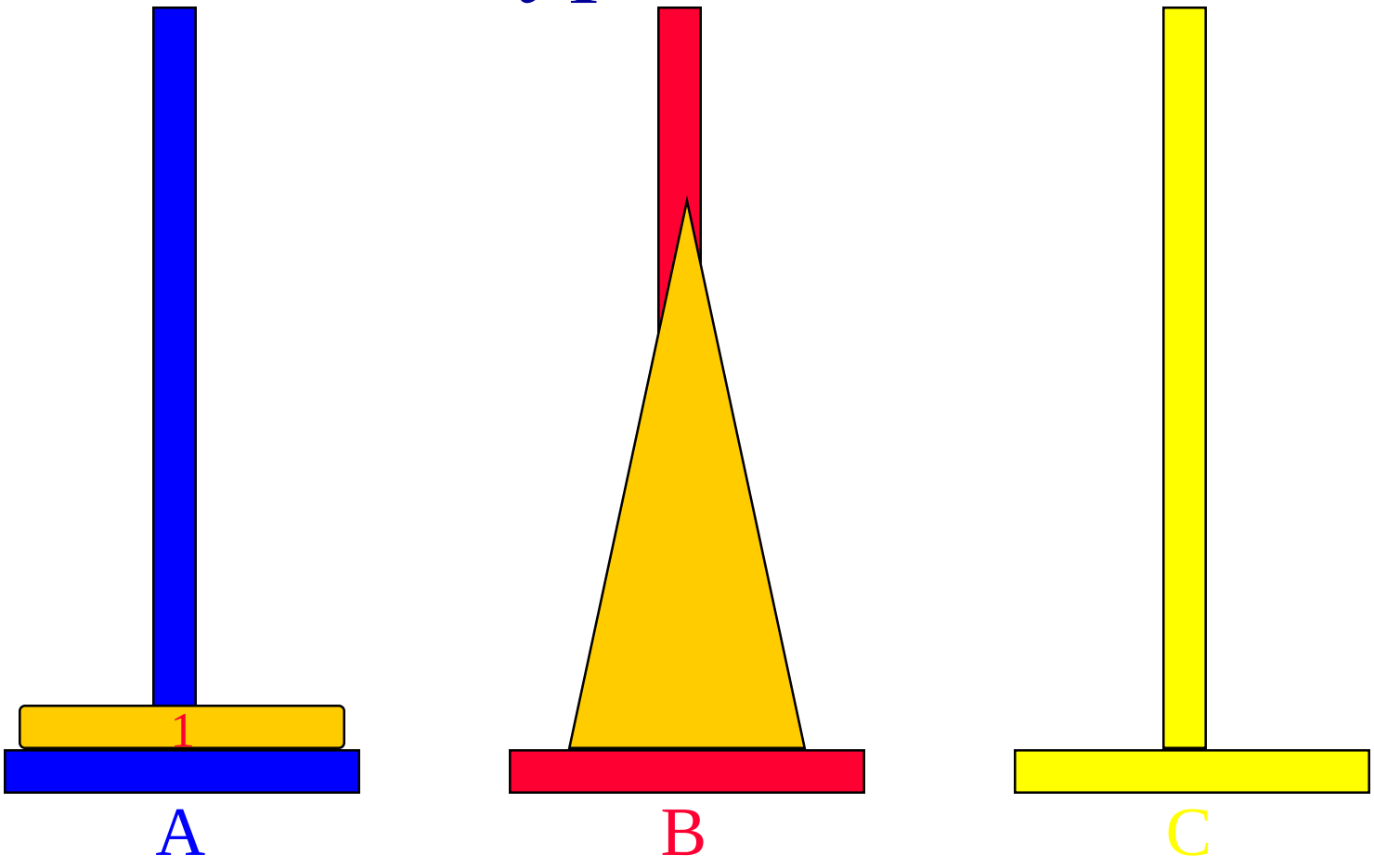
- 3 дисктэй Ханойн цамхаг
- 7 дискийг хөдөлгөлөө

Рекурсив шийдэл



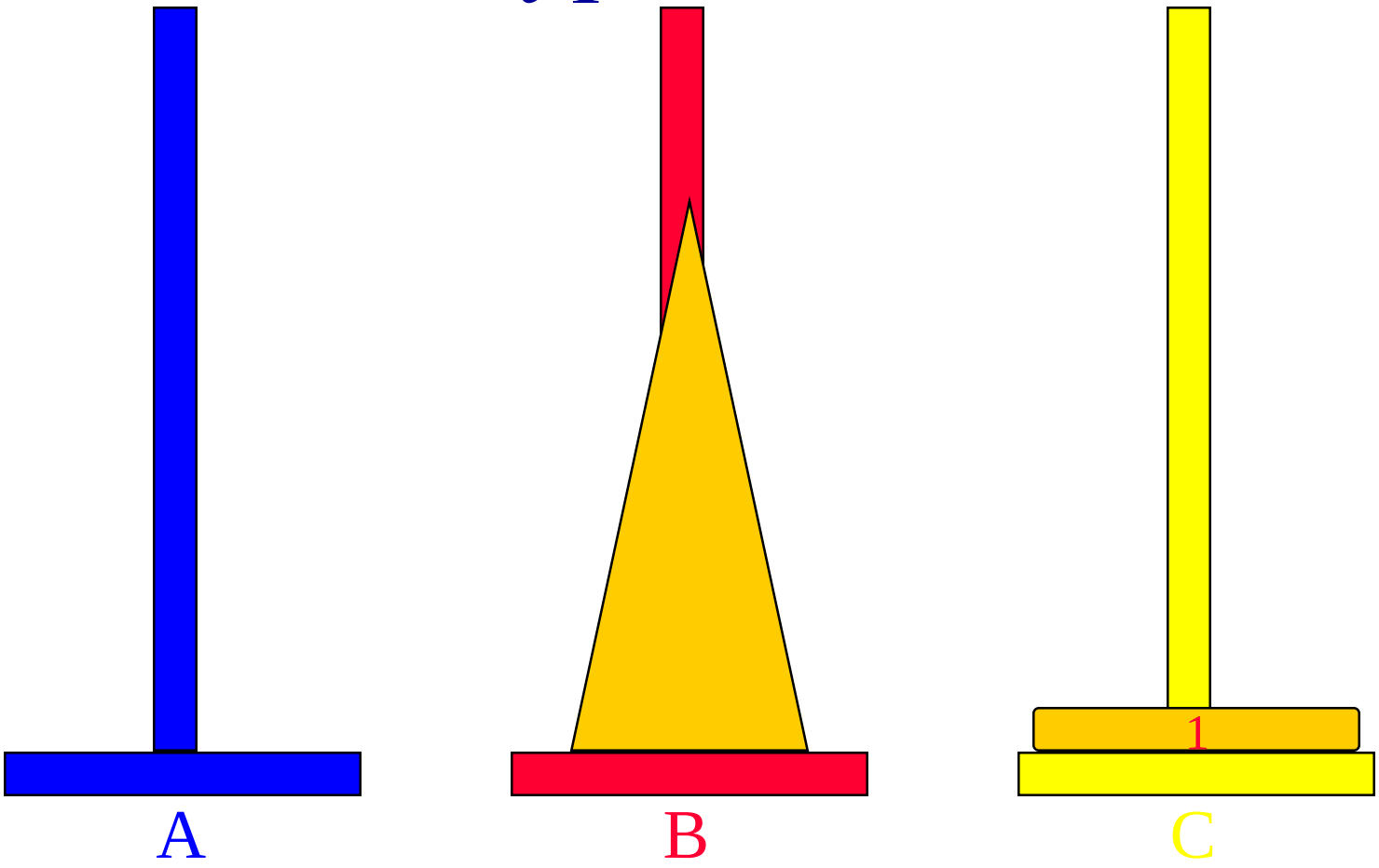
- n > 0 алтан дискийг A -аас C рүү B –г ашиглан шилжүүлнэ
- оройн n-1 дискийг A -аас B рүү C –г ашиглан шилжүүлнэ

Рекурсив шийдэл



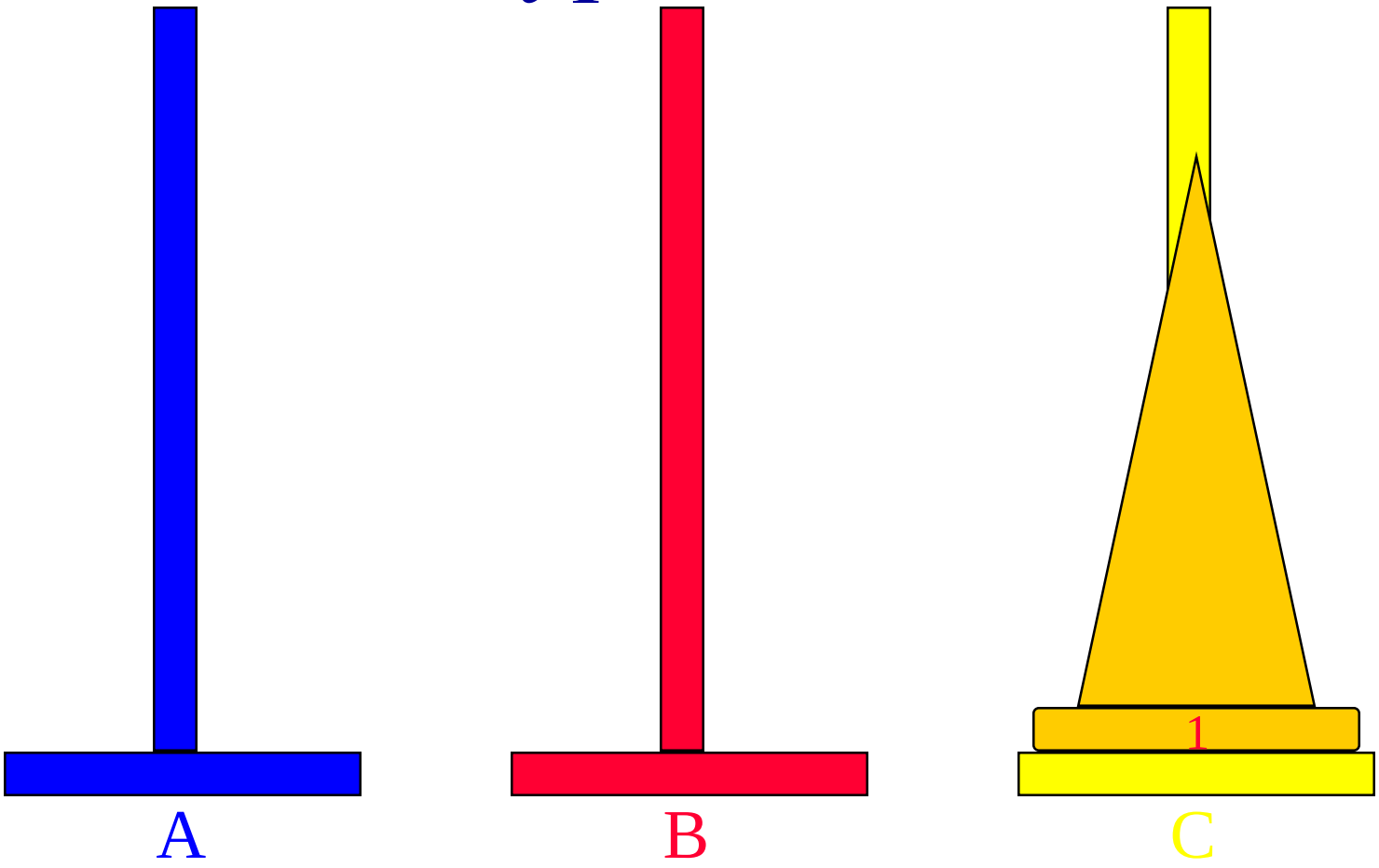
- оройн дискийг A -аас C рүү шилжүүлнэ

Рекурсив шийдэл



- оройн n-1 дискийг B -ээс C рүү A –г ашиглан шилжүүлнэ

Рекурсив шийдэл

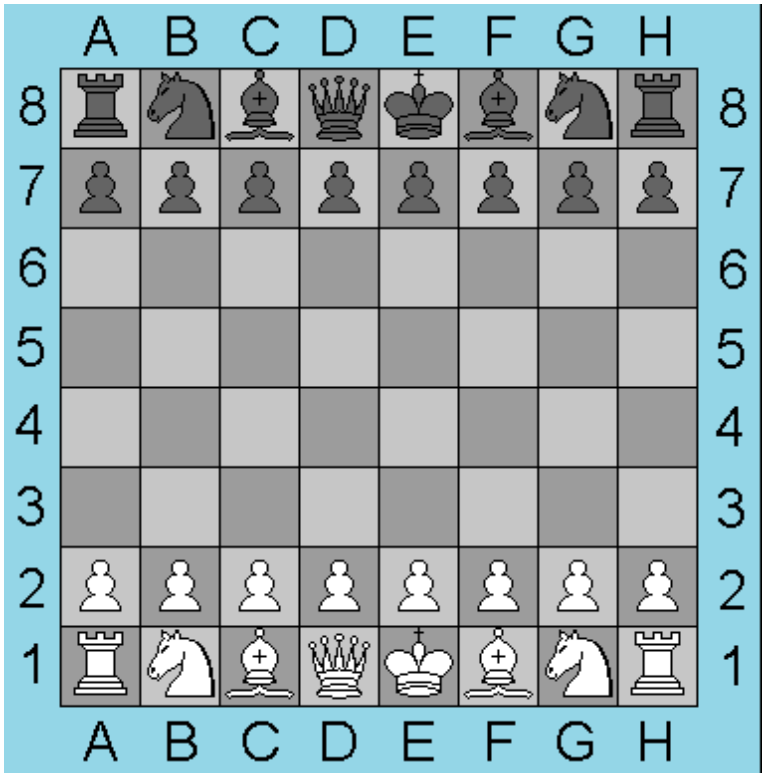


- moves(n) = 0, n = 0 бол
- moves(n) = 2*moves(n-1) + 1 = 2^n-1 , n > 0 бол

Ханойн цамхаг

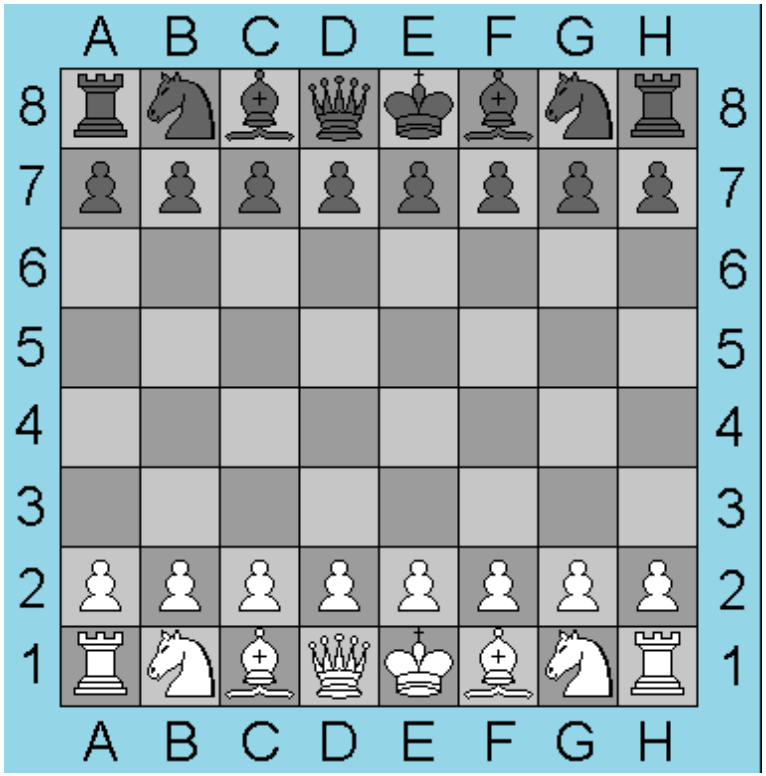
- moves(64) = $1.8 * 10^{19}$ (ойролцоогоор)
- 10^9 шилжүүлэлт/сек хурдтай компьютер 570 орчим жил зарцуулна.
- 1 диск шилжүүлэлт/мин хийдэг лам $3.4 * 10^{13}$ жил зарцуулна.

Шатрын хөлөг



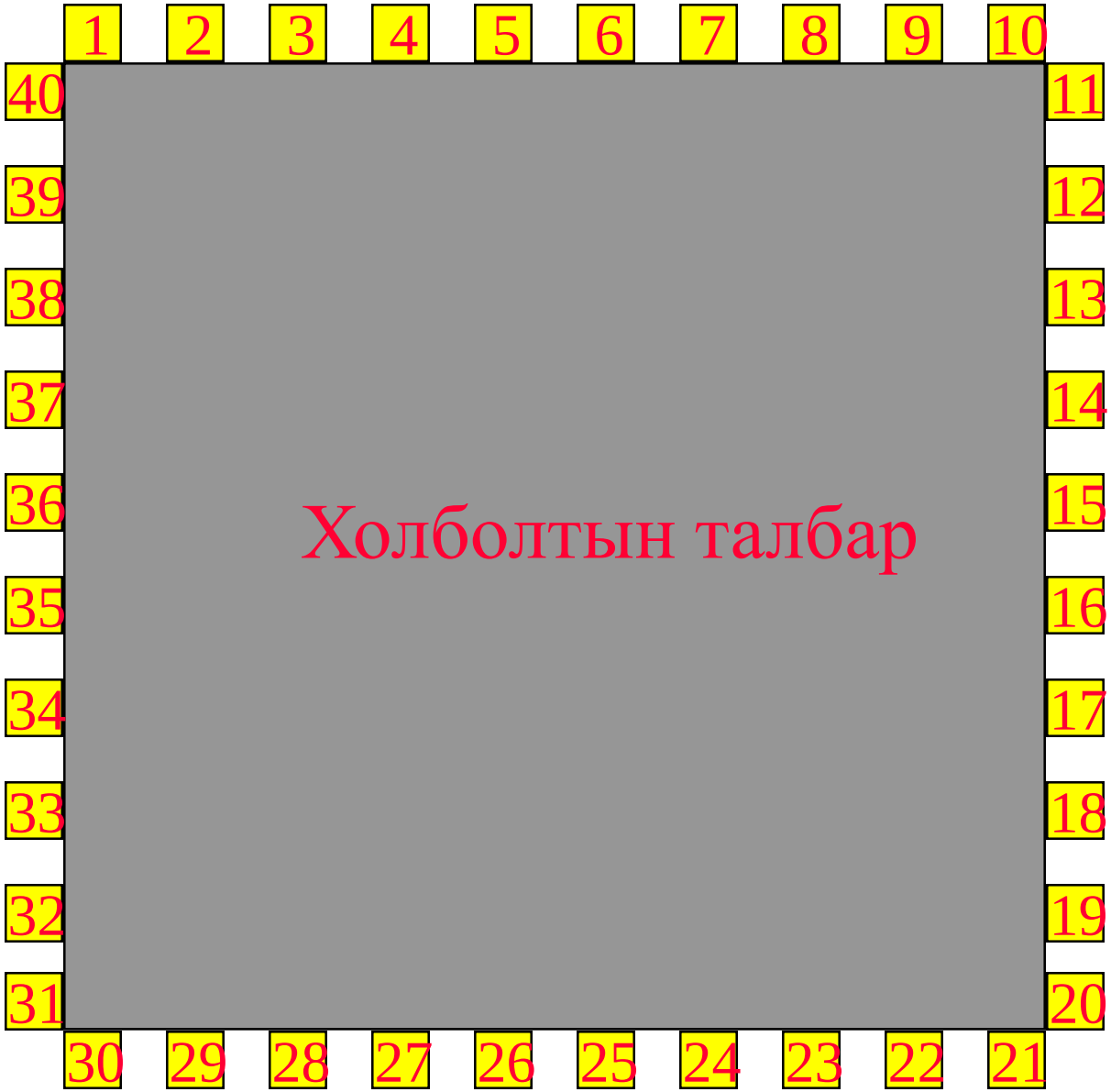
- Эхний нүдэнд 1 будаа, дараагийнхад 2, дараагийнхад 4, гэх мэт.
- Шаардлагатай талбай тэлхийг бүрхэнэ.

Шатрын хөлөг



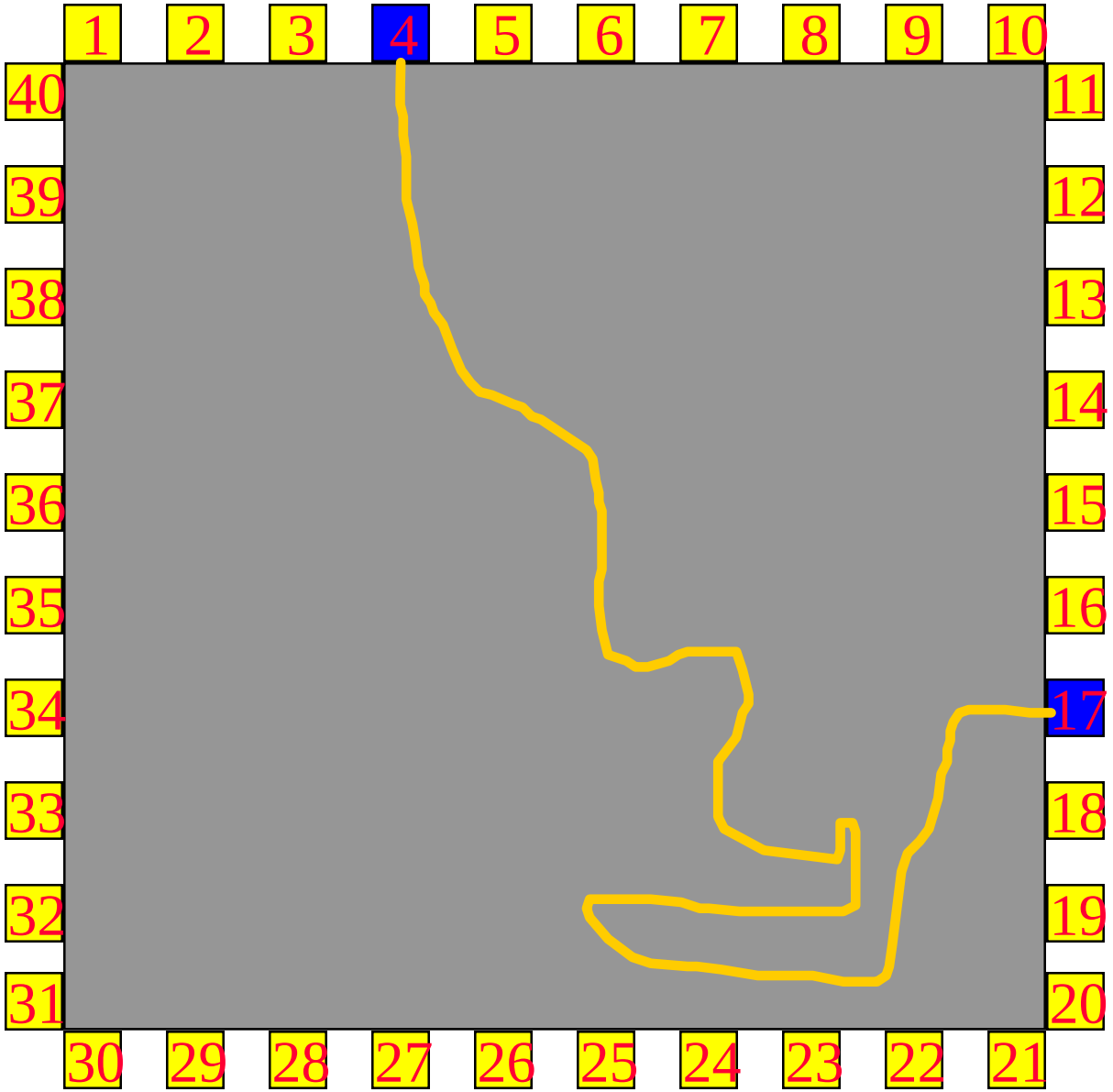
- Эхний нүдэнд 1 цент, дараагийнхад 2, дараагийнхад 4, гэх мэт.
- $3.6 * 10^{17} (холбооны төсөв ~ $2 * 10^{12}) .

Холболтын хайрцаг



2 цэгийн холболт

1-3 , 18-40 холболт зүүн доор хийгдэнэ.



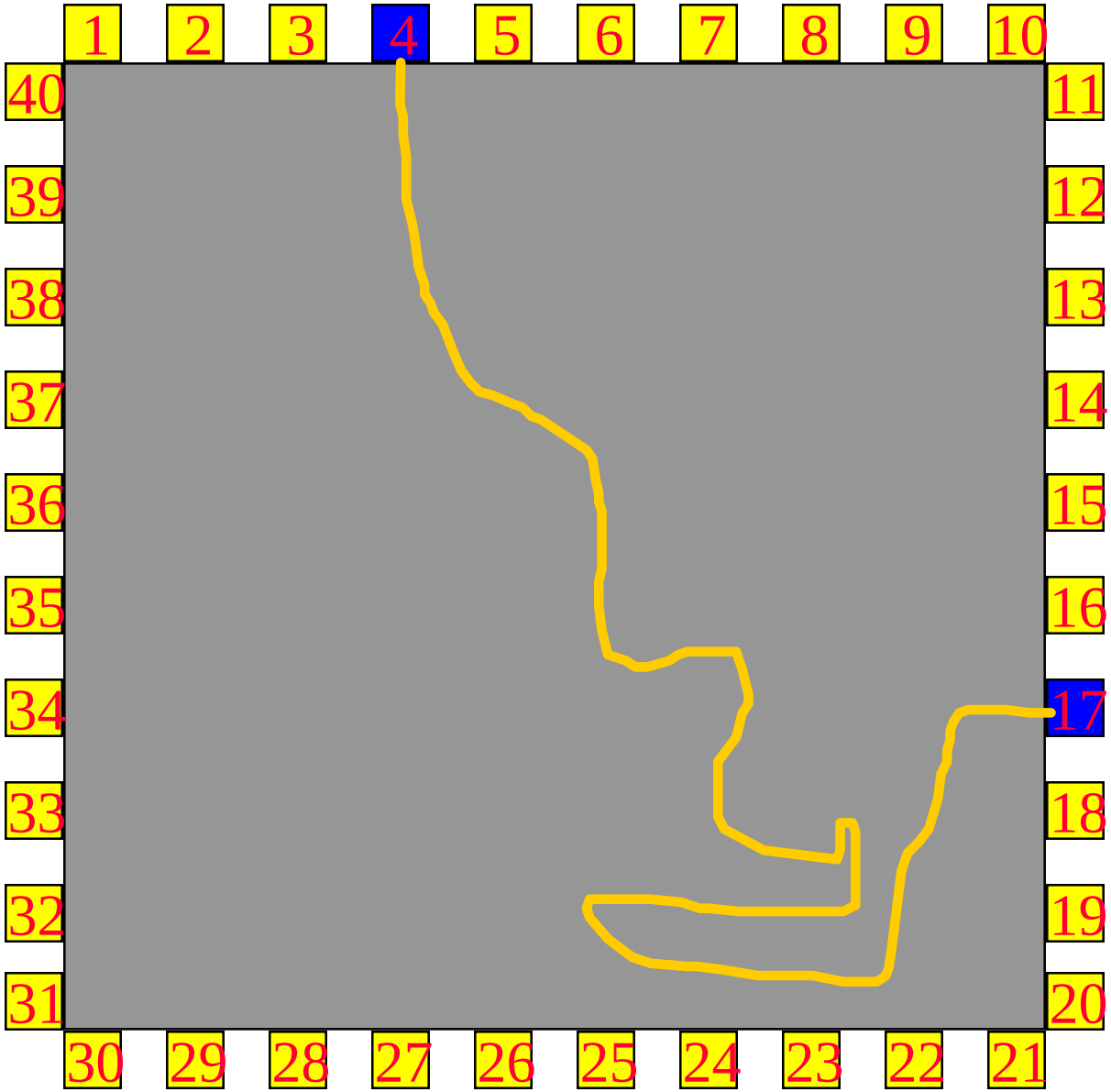
5-16 баруун дээр хийгдэнэ.

2 цэгийн холболт

(u,v), u<v бол 2-цэгийн холболт.

u эхлэх цэг.

v төгсөх цэг.

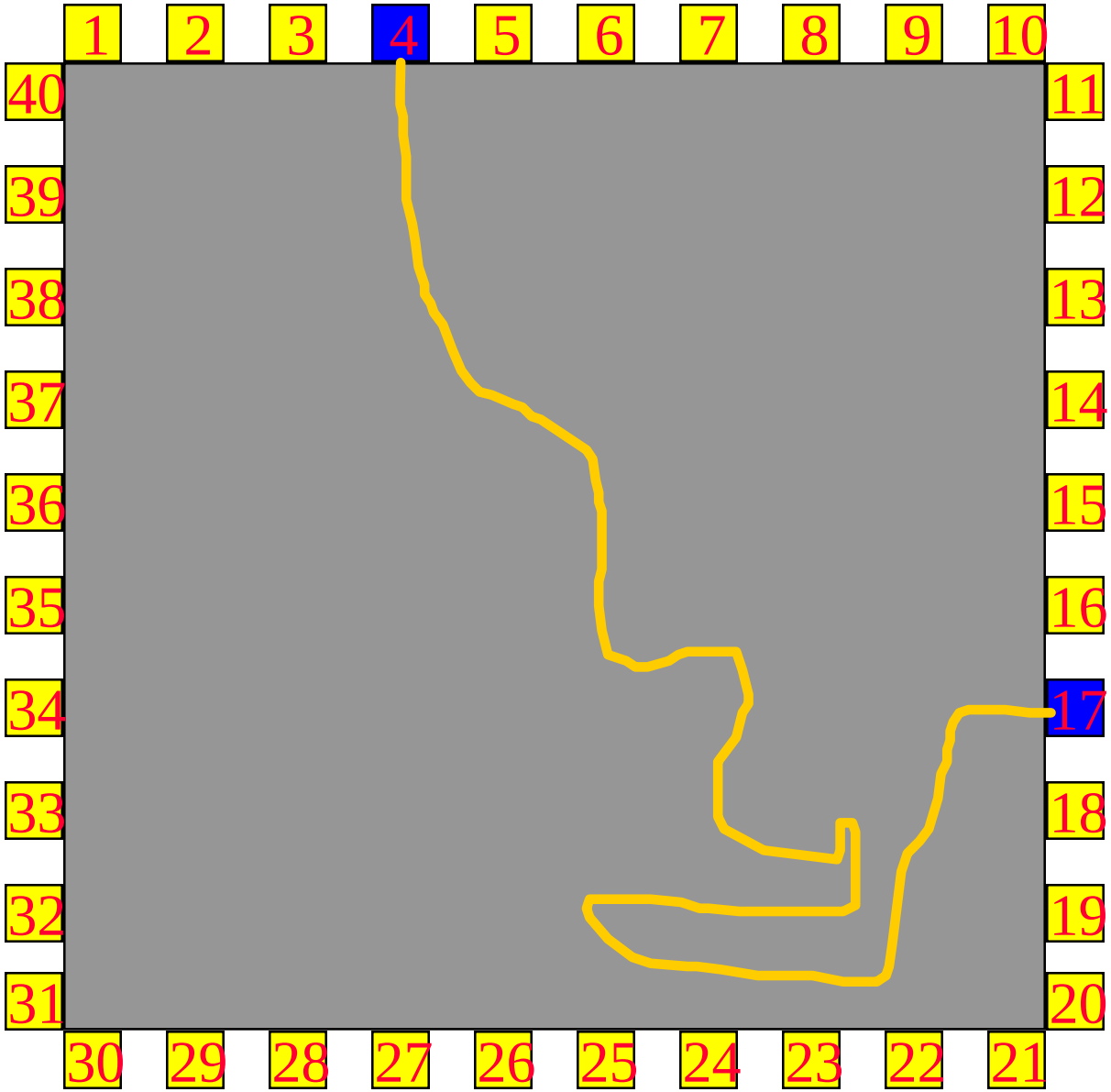


Цэгүүдийг цагын зүүний дагуу 1 -ээс эхлэн дугаарлав

2 цэгийн холболт

Эхлэх цэгийг => стект хийнэ.

Төгсөх цэгт => харгалзах эхлэх цэг стекын оройд байна.



Аргыг Дуудах ба Буцах

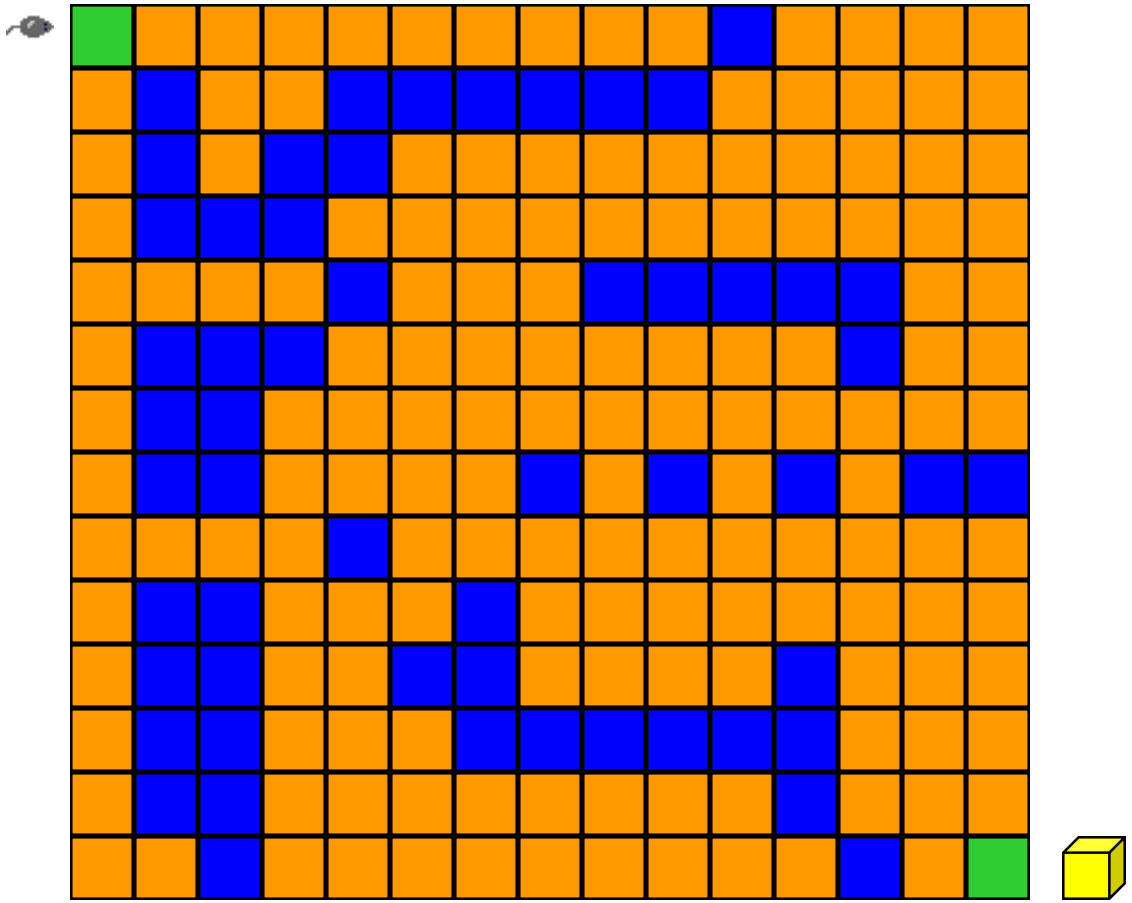
```
public void a()
{ …; b(); …}
public void b()
{ …; c(); …}
public void c()
{ …; d(); …}
public void d()
{ …; e(); …}
public void e()
{ …; c(); …}
```

буцах хаяг d()
буцах хаяг c()
буцах хаяг e()
буцах хаяг d()
буцах хаяг c()
буцах хаяг b()
буцах хаяг a()

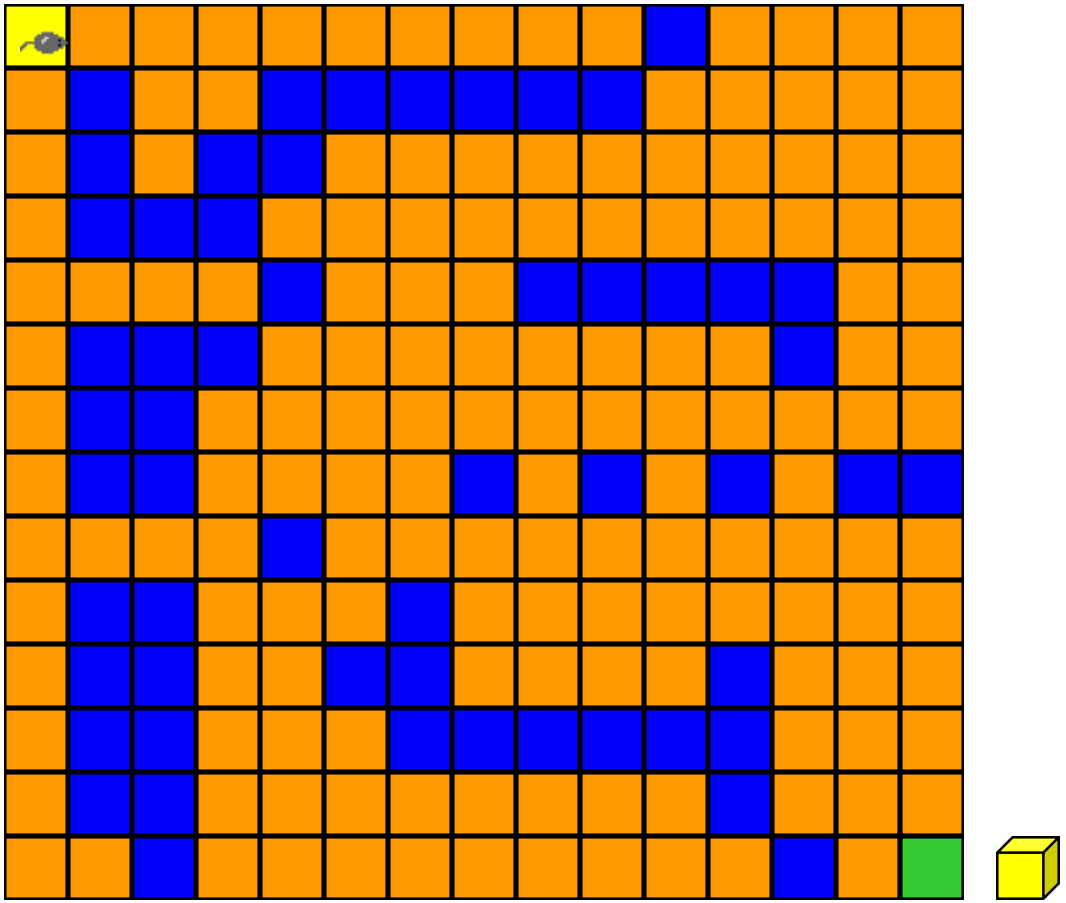
Try-Throw-Catch

- try блокт ороход, энэ блокын хаягийг стект хийнэ.
- Онцгой тохиолдол унахад, стекын оройд байгаа try блокын хаягийг авна(стек хоосон бол зогсоно).
- Гаргаж авсан try блокт харгалзах catch блок байхгүй бол, өмнөх алхам руу буцна.
- Гаргаж авсан try блокт харгалзах catch блок байгаа бол, тэр catch блок хэрэгжинэ.

Төөрөлдсөн оготно

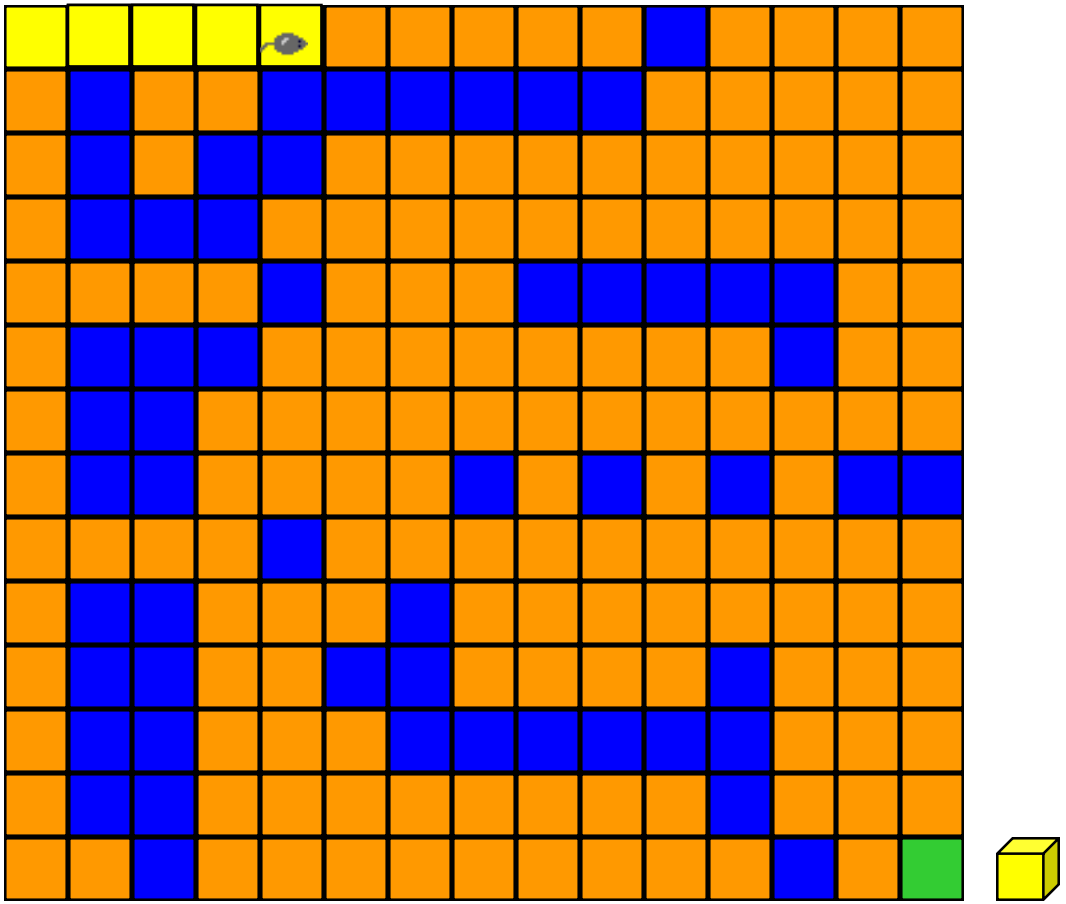


Төөрөлдсөн оготно



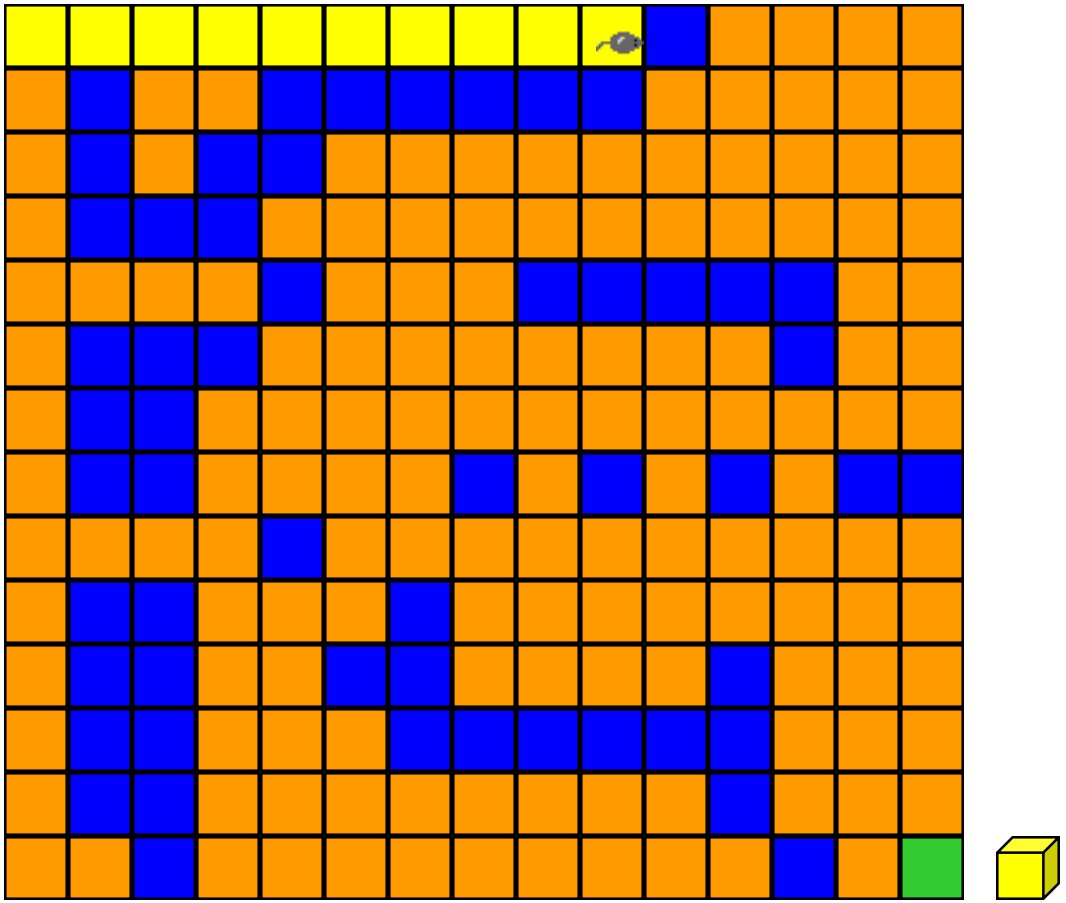
- Явах чиглэл: right, down, left, up
- Дахин орохгүйн тулд нүдийг хаана.

Төөрөлдсөн оготно



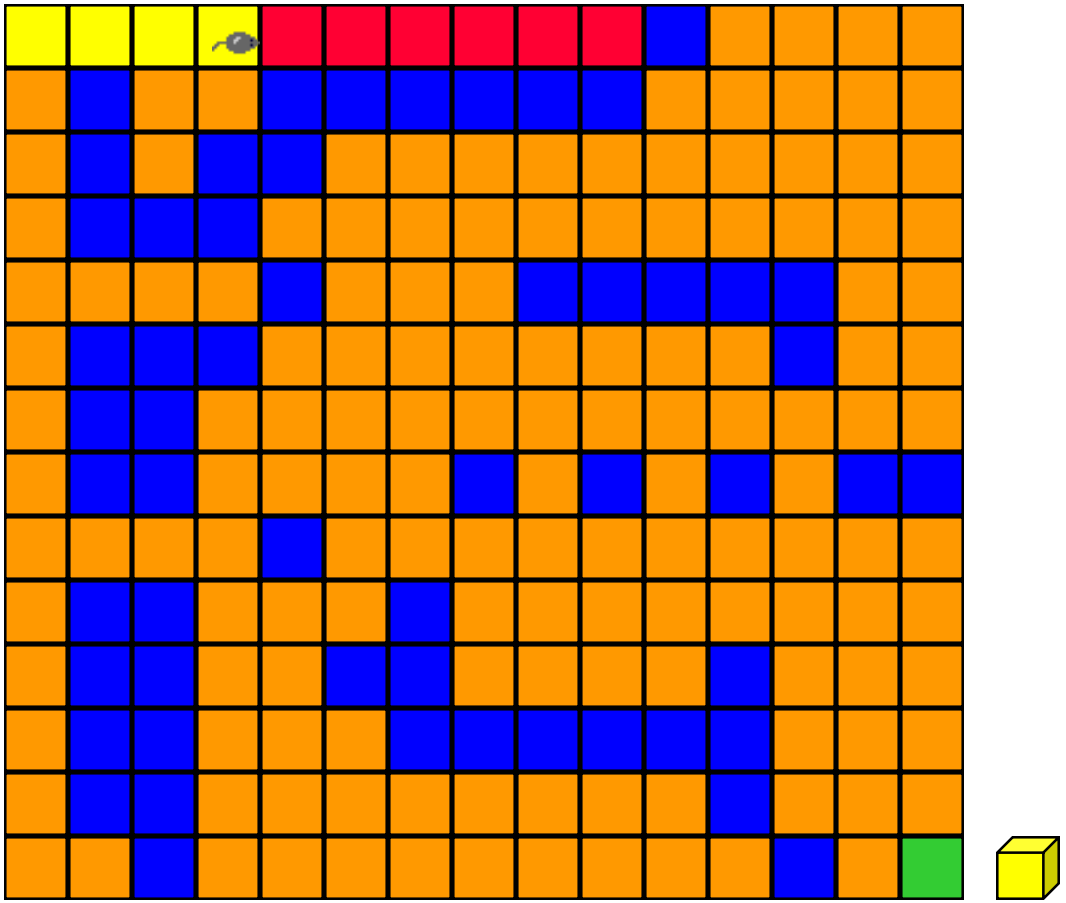
- Явах чиглэл: right, down, left, up
- Дахин орохгүйн тулд нүдийг хаана.

Төөрөлдсөн оготно



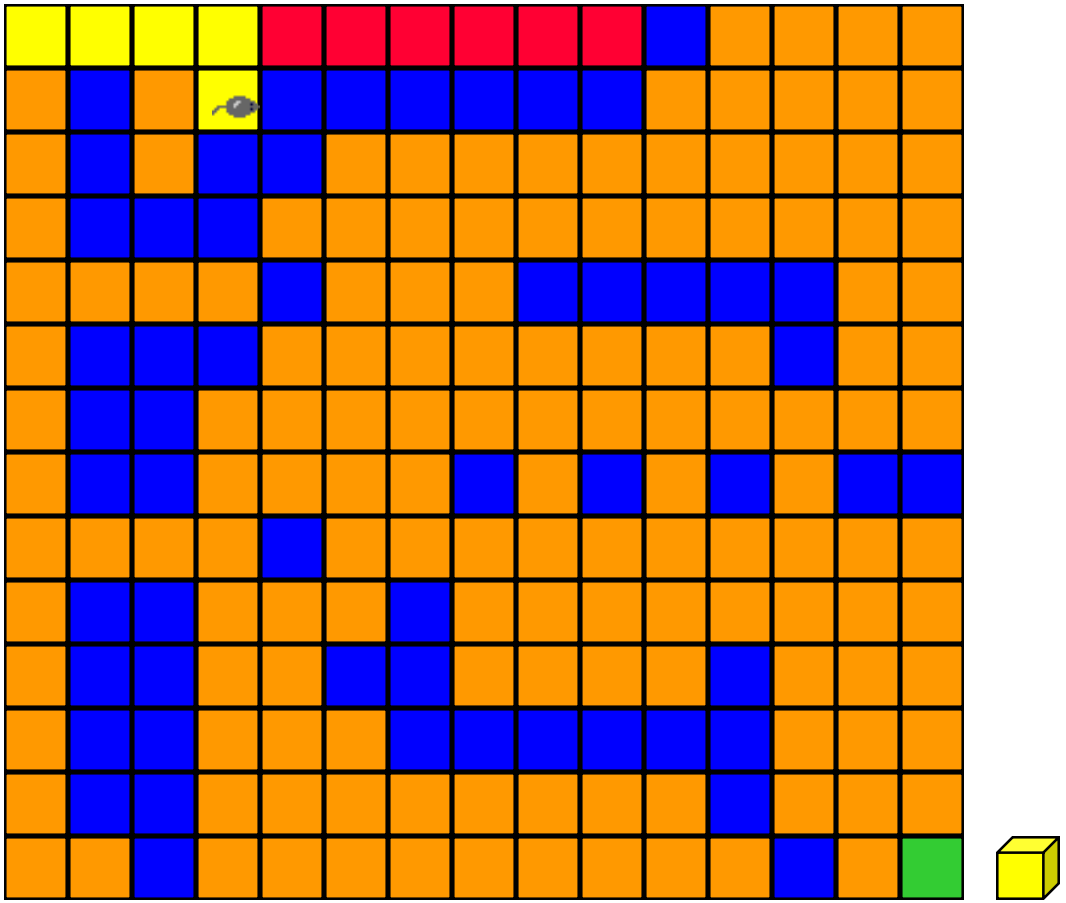
- Урагш явах боломжтой нүд хүртэл ухрах.

Төөрөлдсөн оготно



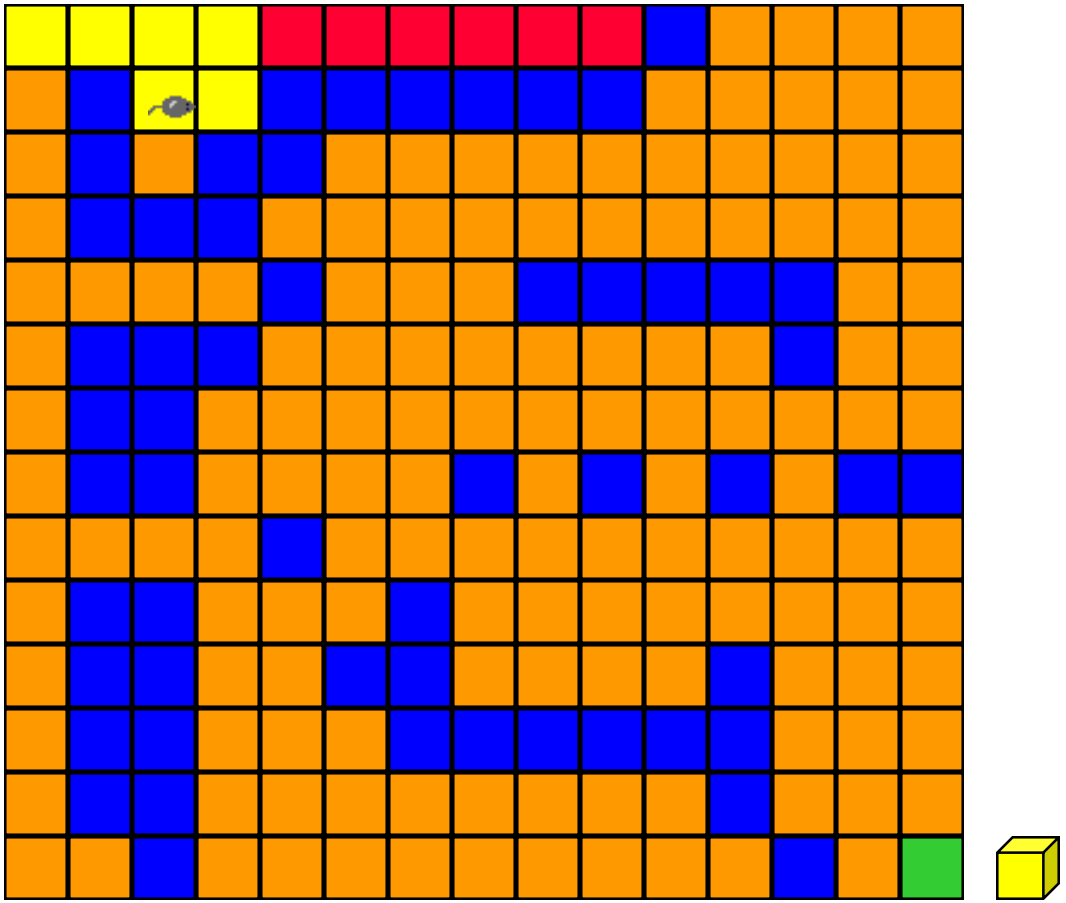
- Доош явах.

Төөрөлдсөн оготно



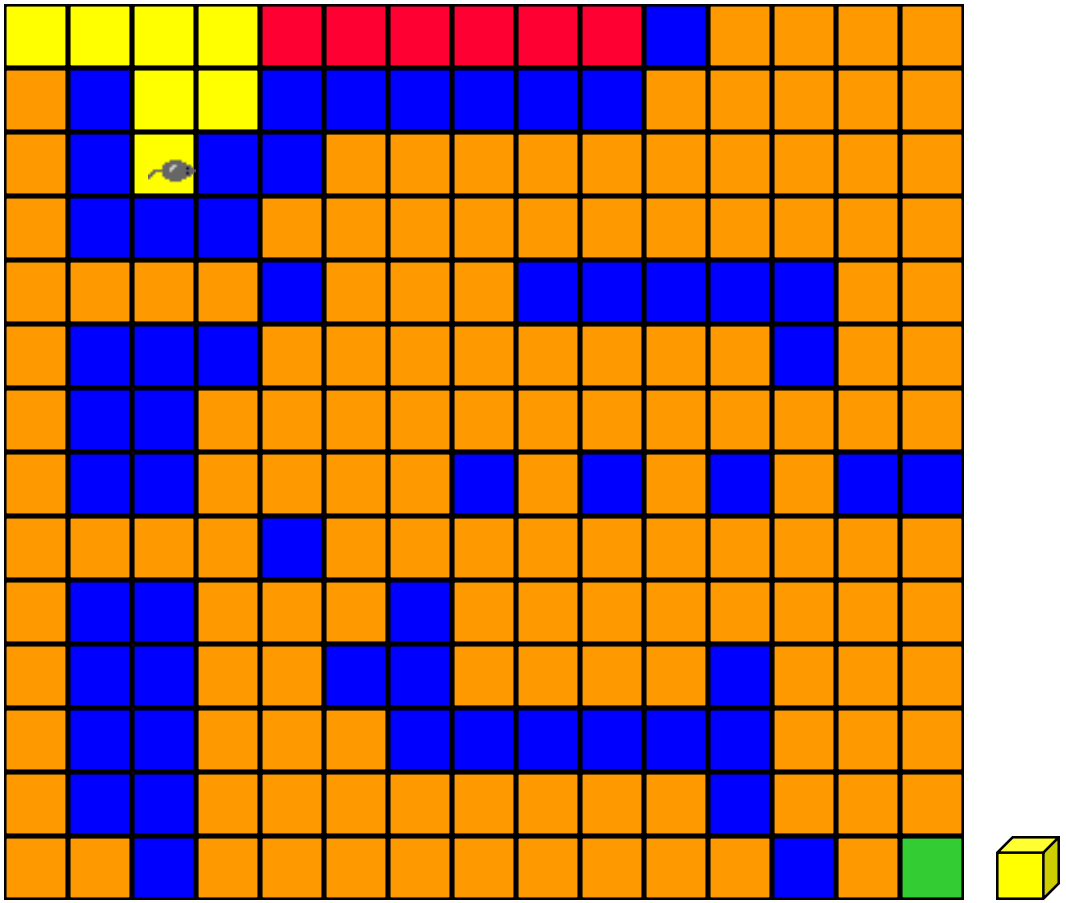
- Зүүн тийш явах.

Төөрөлдсөн оготно



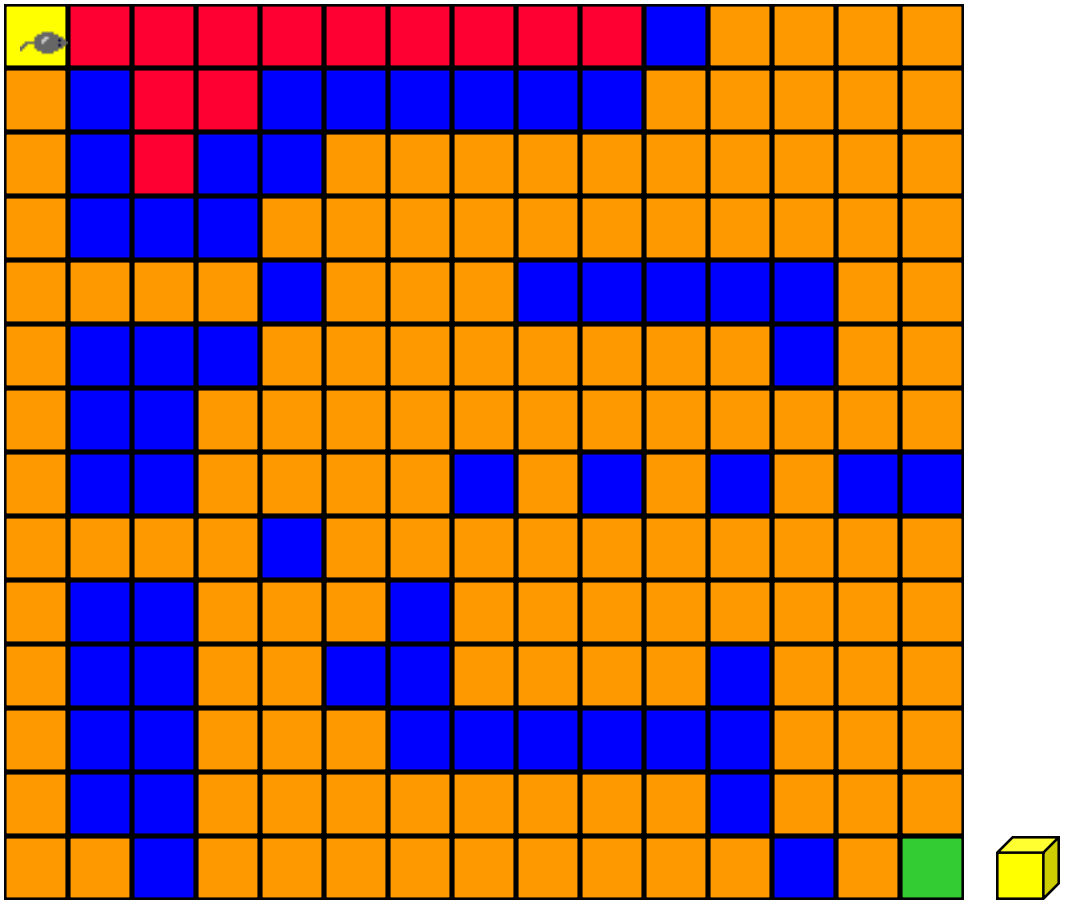
- Доош явах.

Төөрөлдсөн оготно



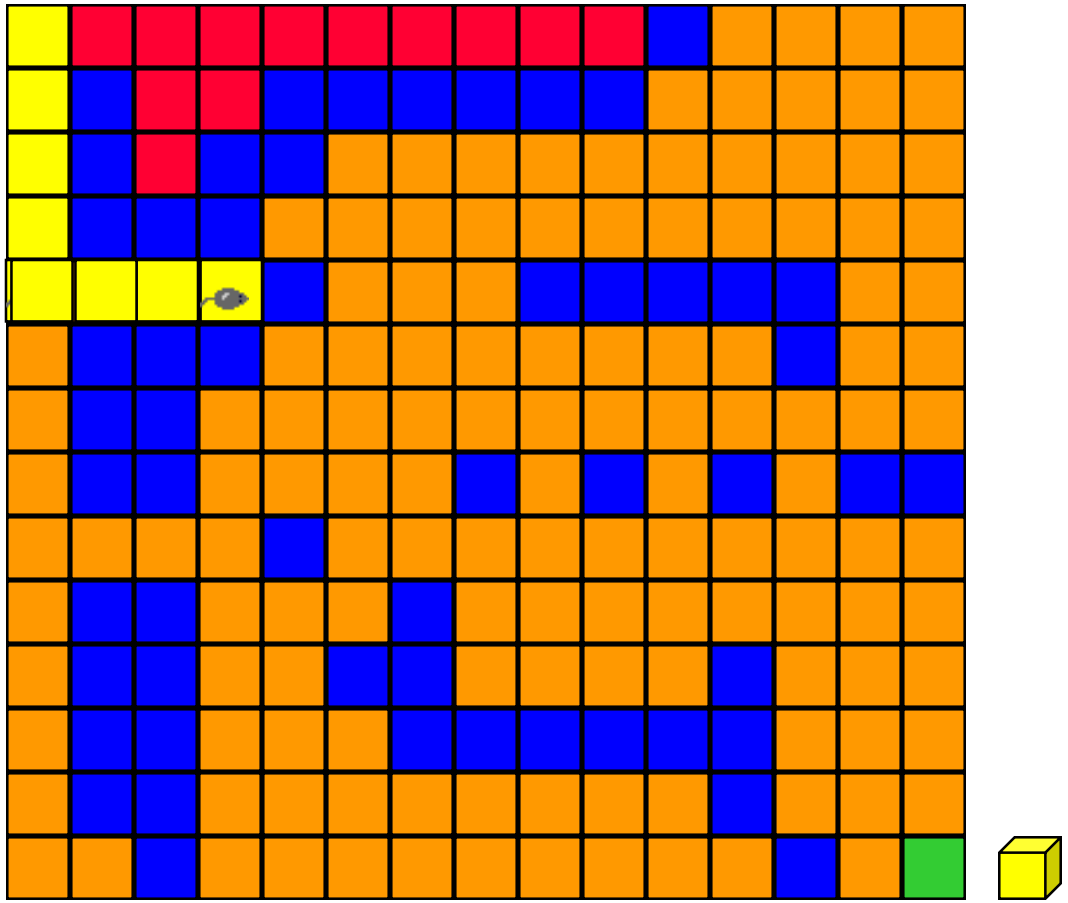
- Урагш явах боломжтой нүд хүртэл ухрах.

Төөрөлдсөн оготно



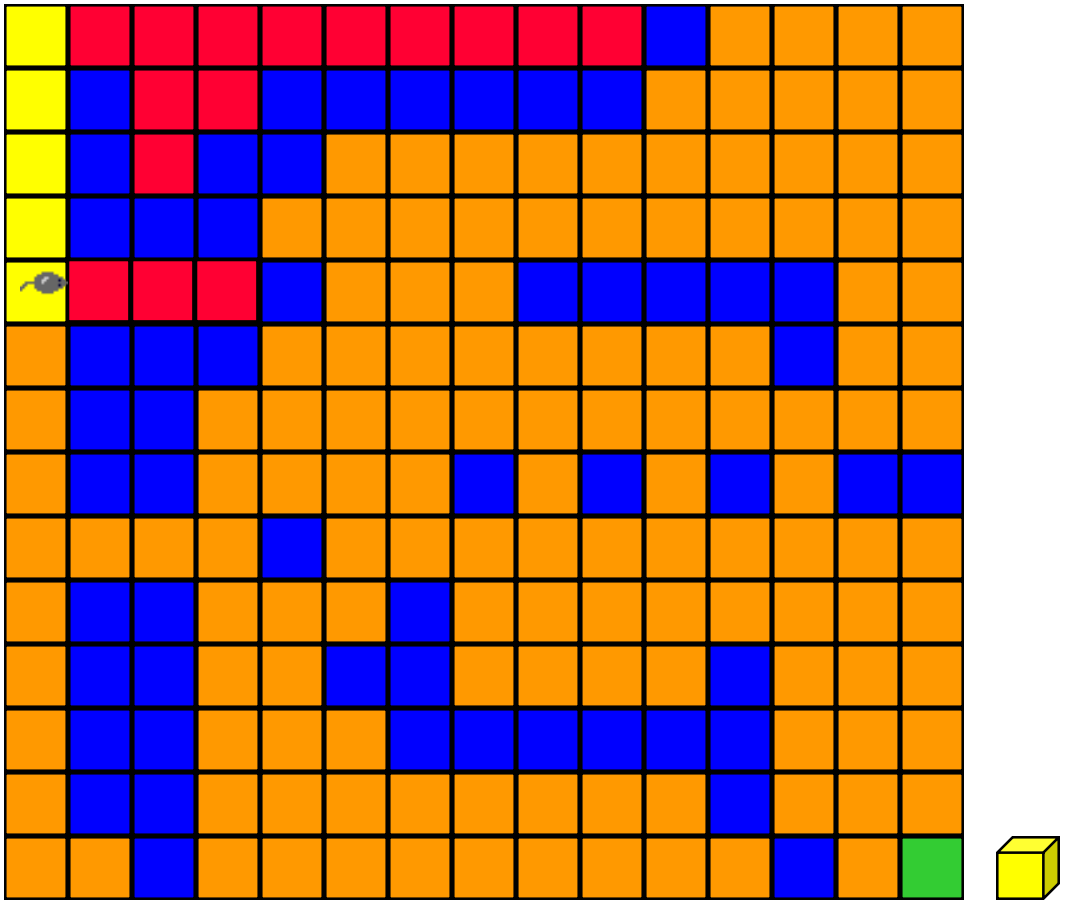
- Урагш явах боломжтой нүд хүртэл ухрах.
- Доош явах.

Төөрөлдсөн оготно



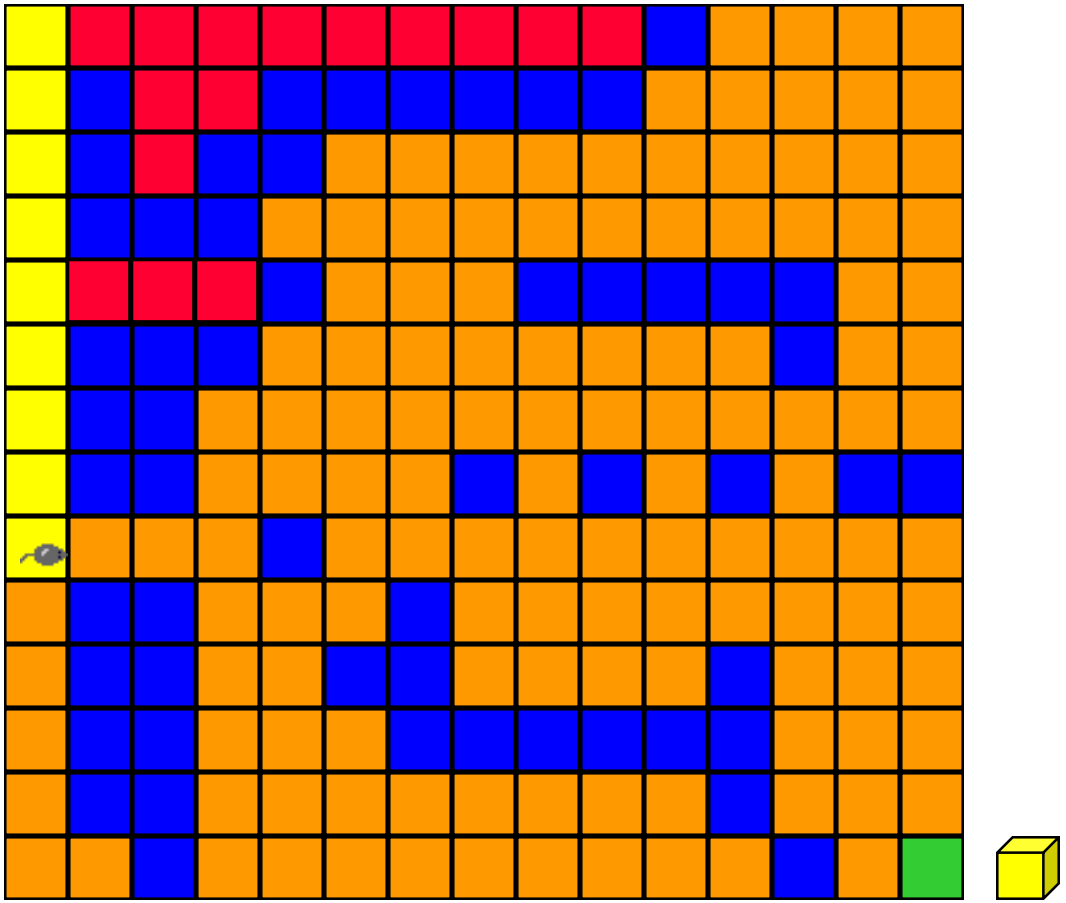
- Баруун тийш явах.
- Буцах.

Төөрөлдсөн оготно



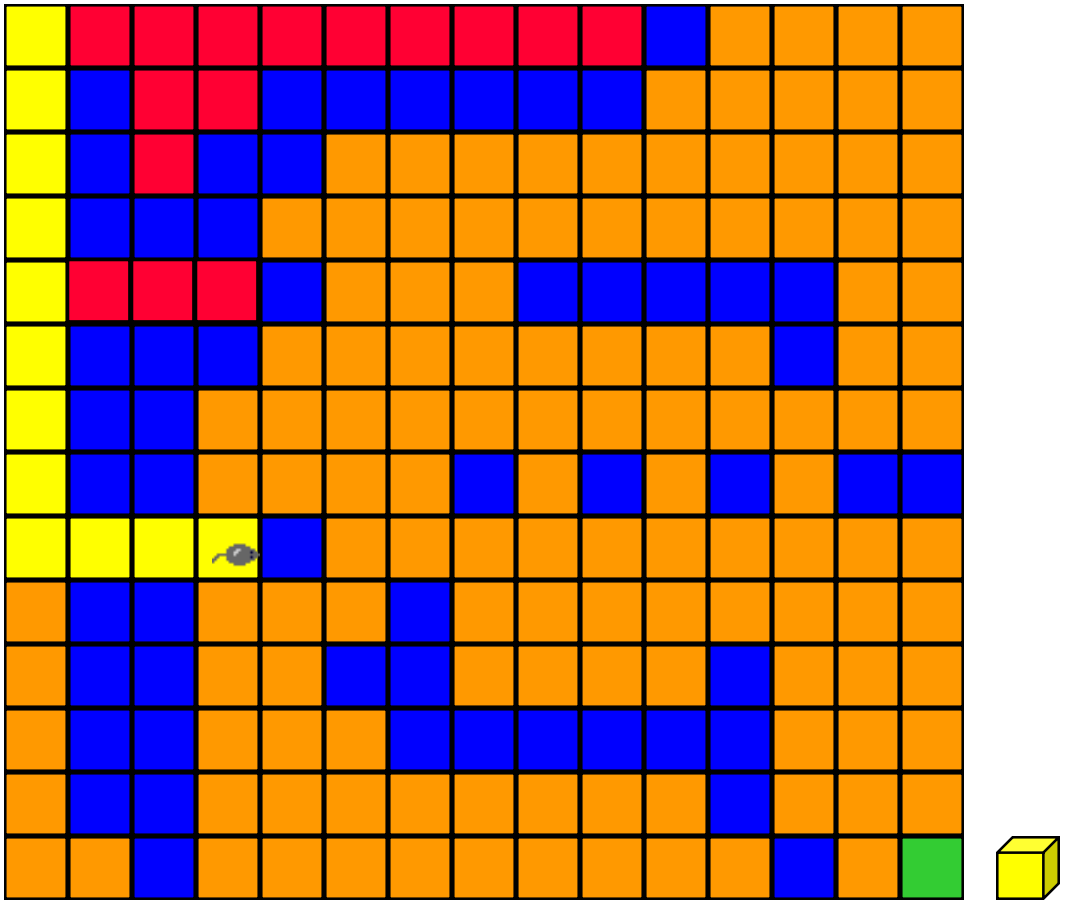
- Доош явах.

Төөрөлдсөн оготно



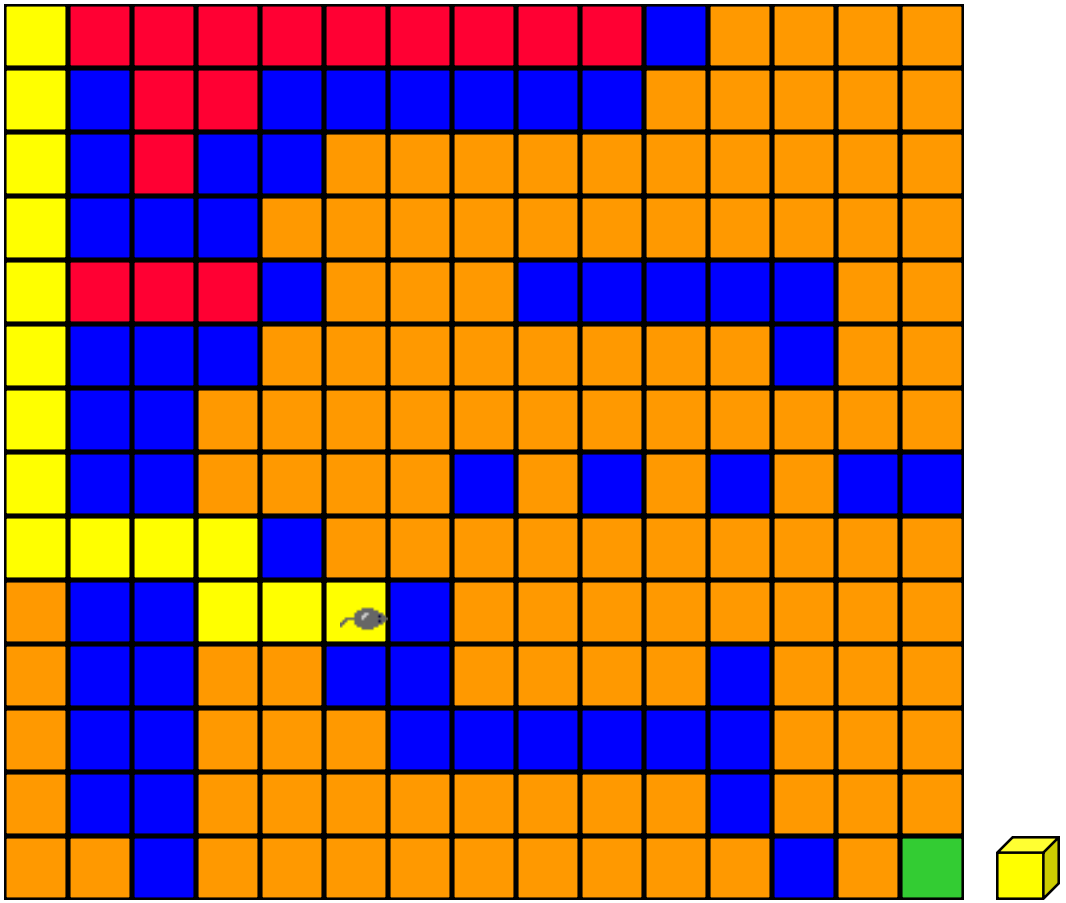
- Баруун тийш явах.

Төөрөлдсөн оготно



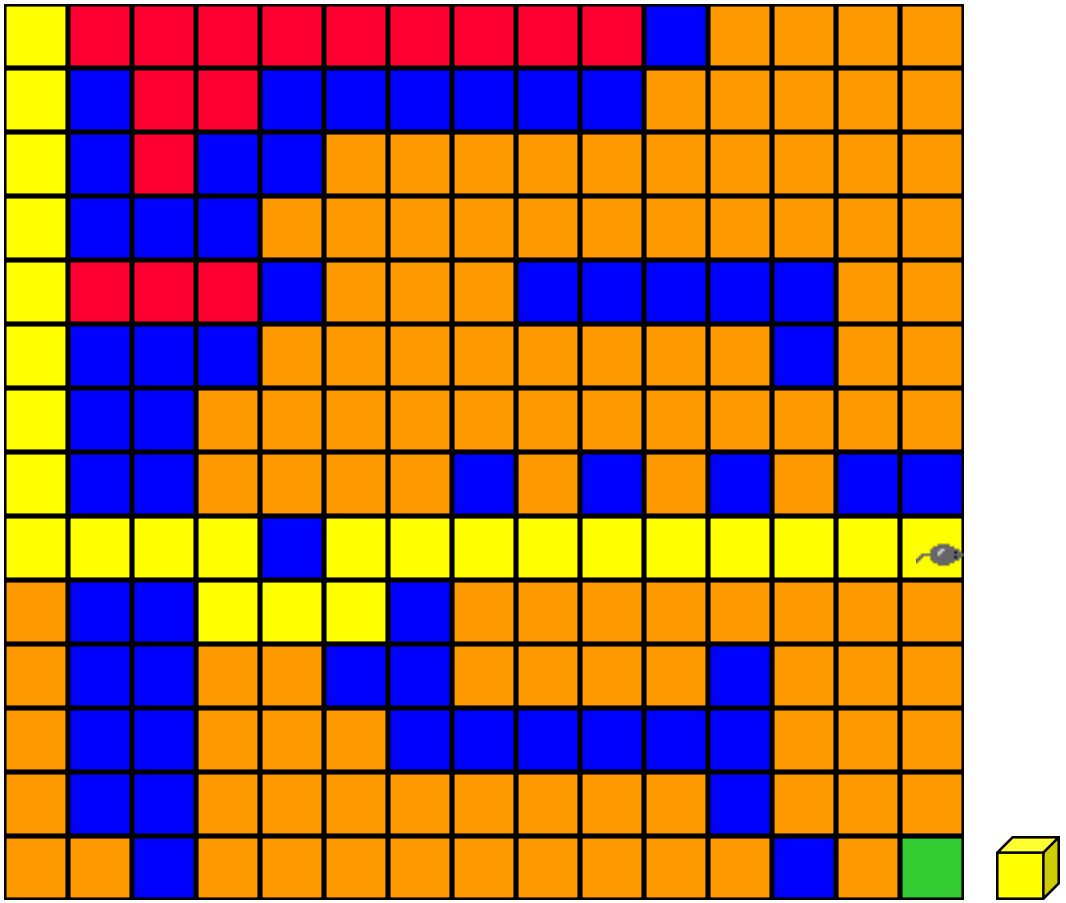
- Нэг доошлоод баруун тийш явах.

Төөрөлдсөн хулгана



- Нэг дээшлээд баруун тийш явах.

Төөрөлдсөн оготно



- Доош явж гарангаа бяслагийг идэх.
- Оготны орсон нүднээс тухайн байршил хүртэлх зам стектэй адилхан ажиллана.