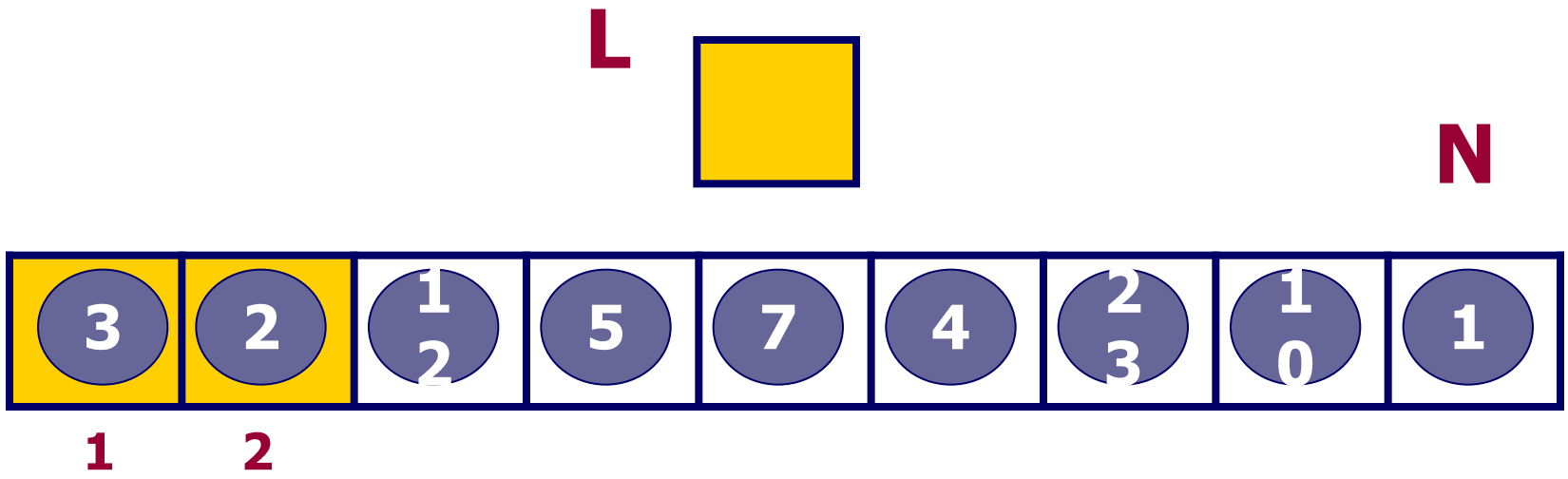
МЕТОДЫ СОРТИРОВКИ МАССИВОВ

СОРТИРОВКА МЕТОДОМ «ПУЗЫРЬКА»

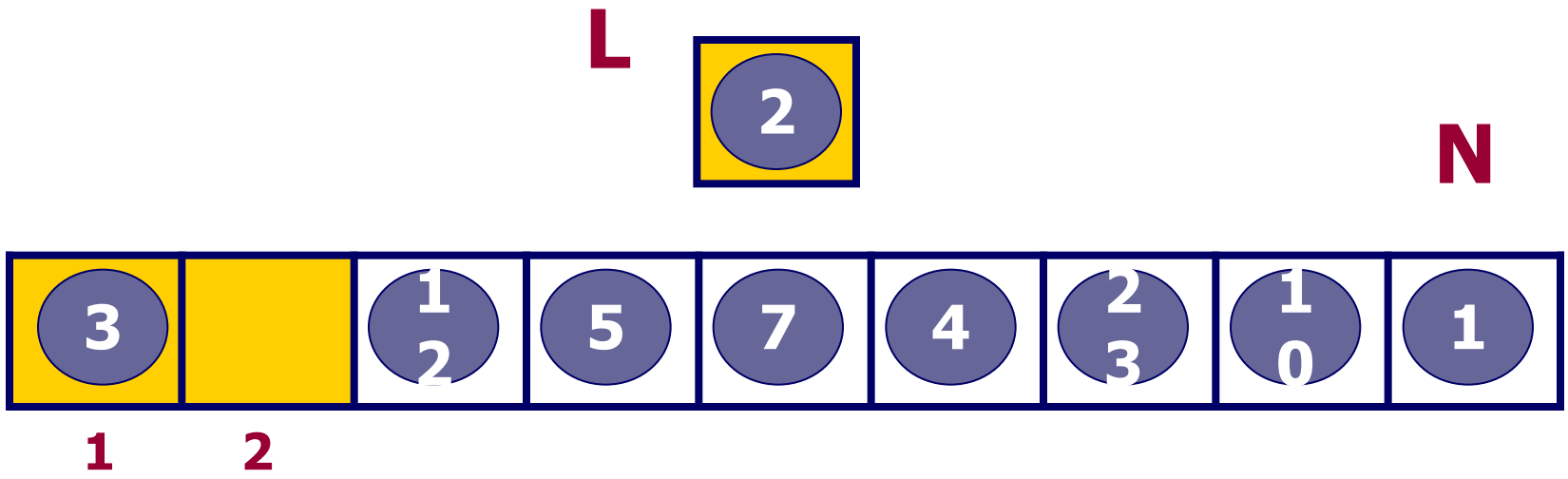
Кондраткова
Татьяна Алексеевна

ГБОУ Лицей № 82 СПб

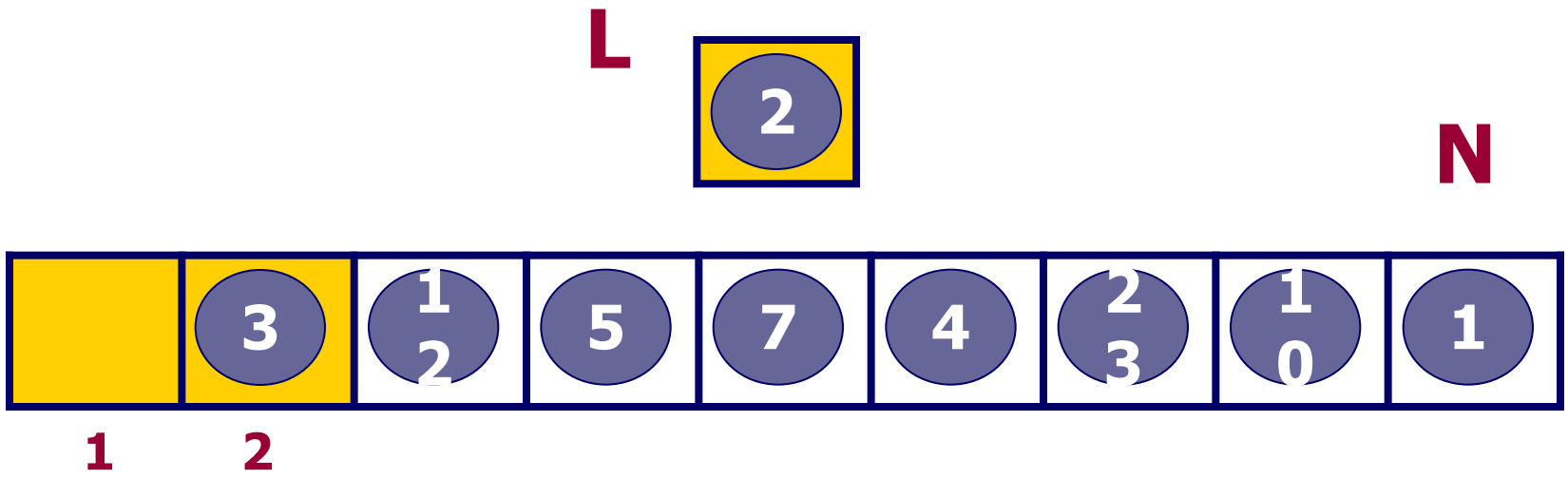
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



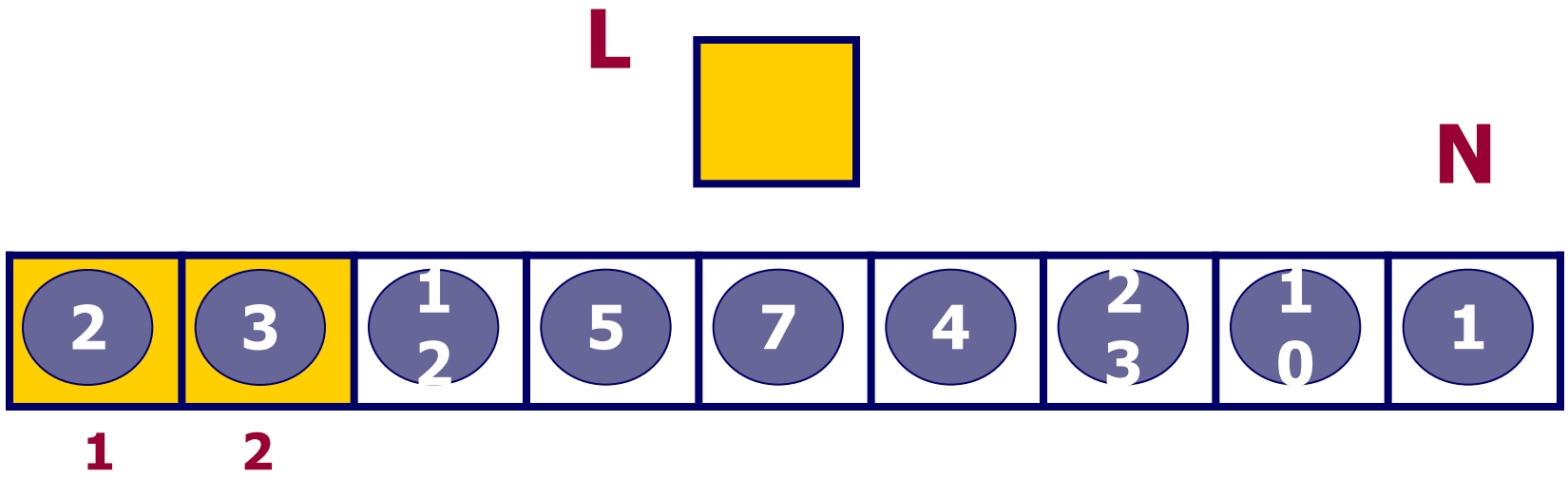
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



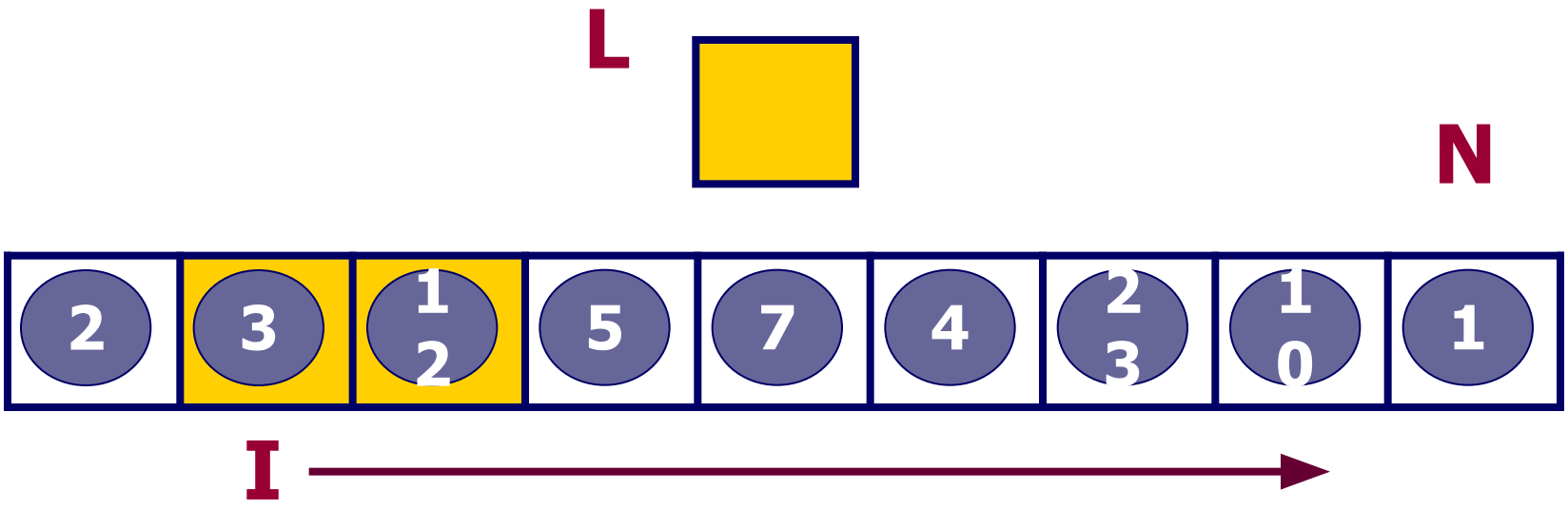
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



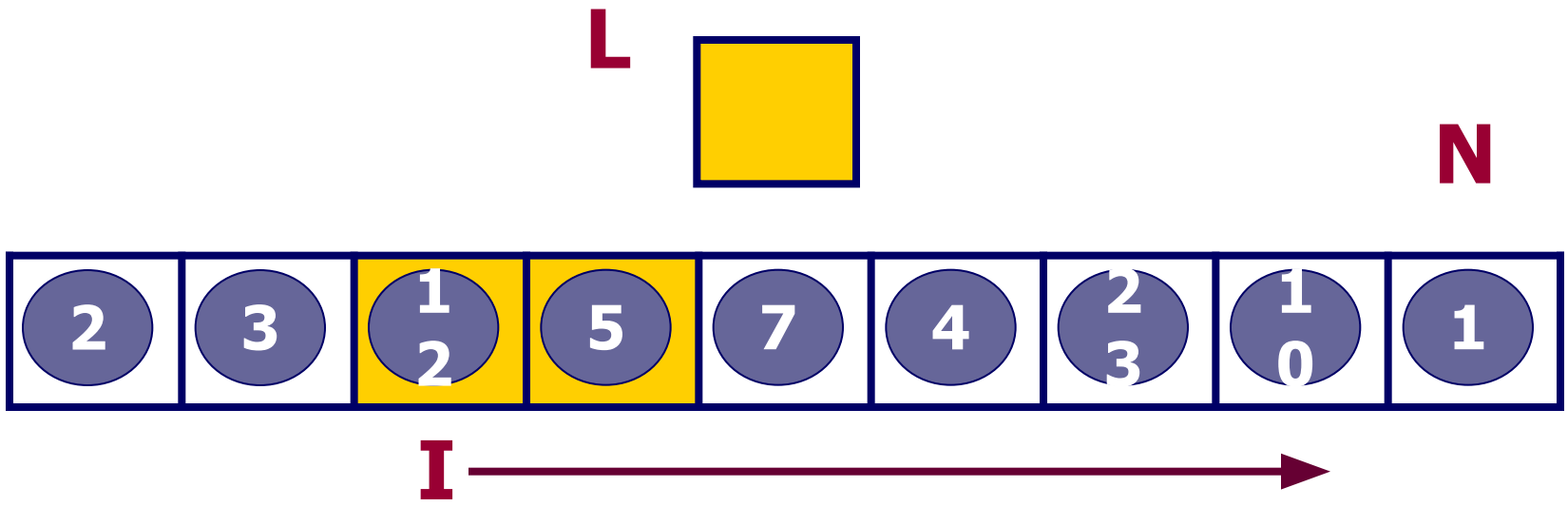
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



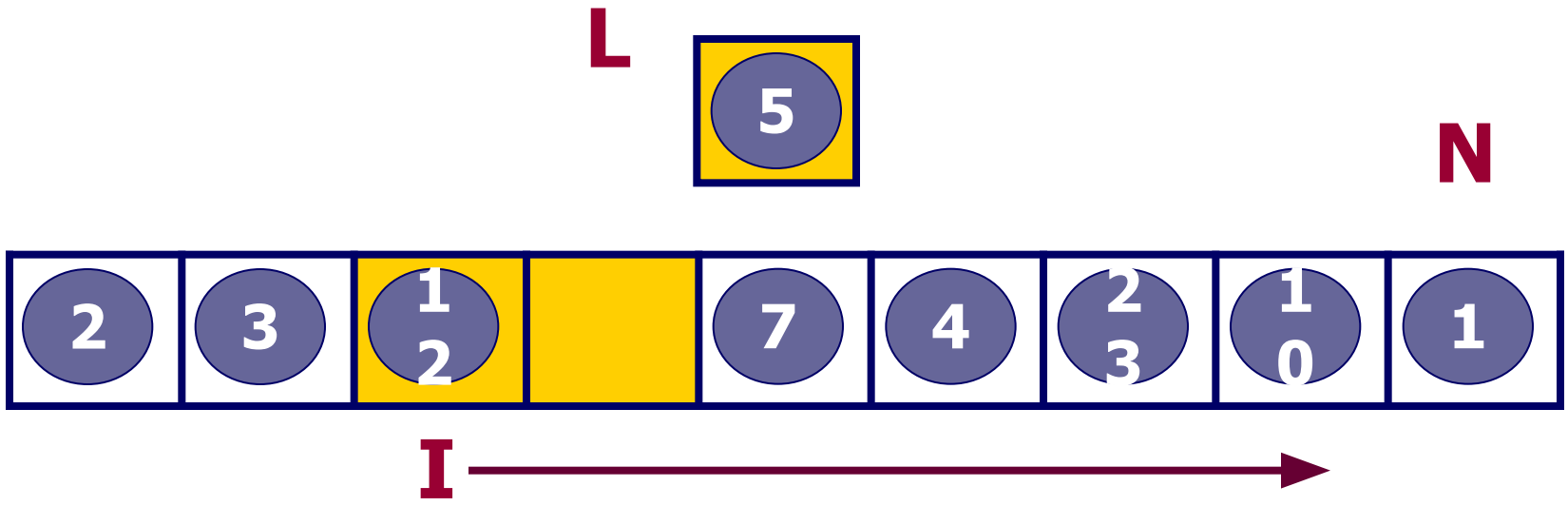
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



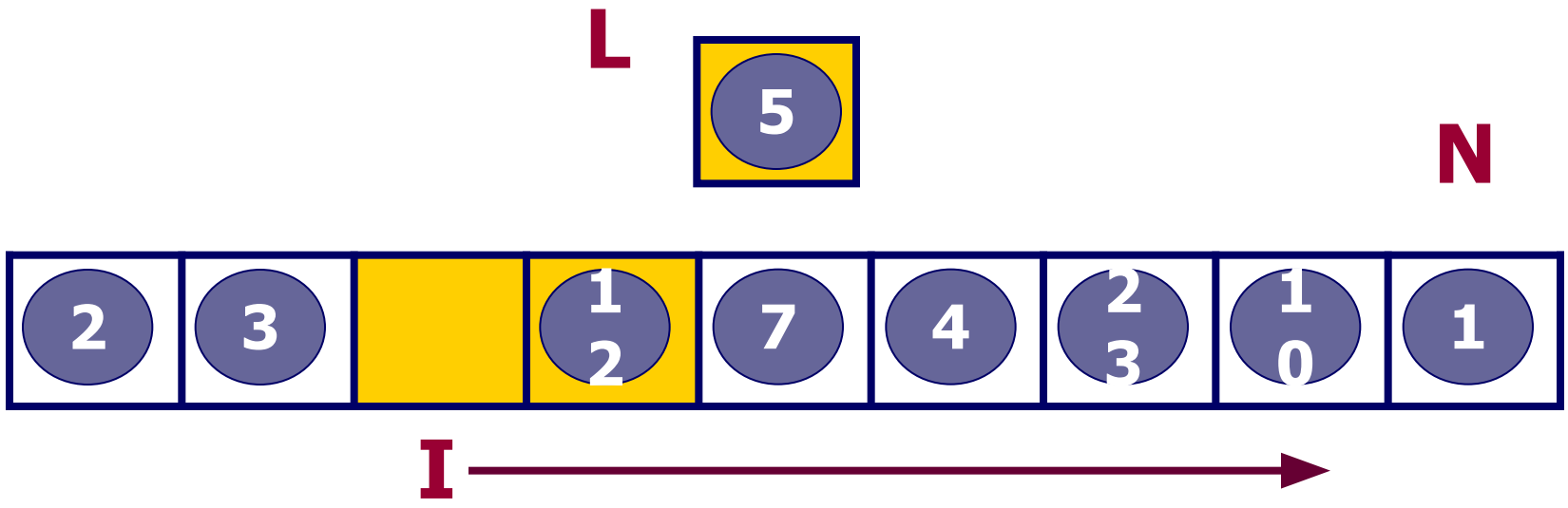
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



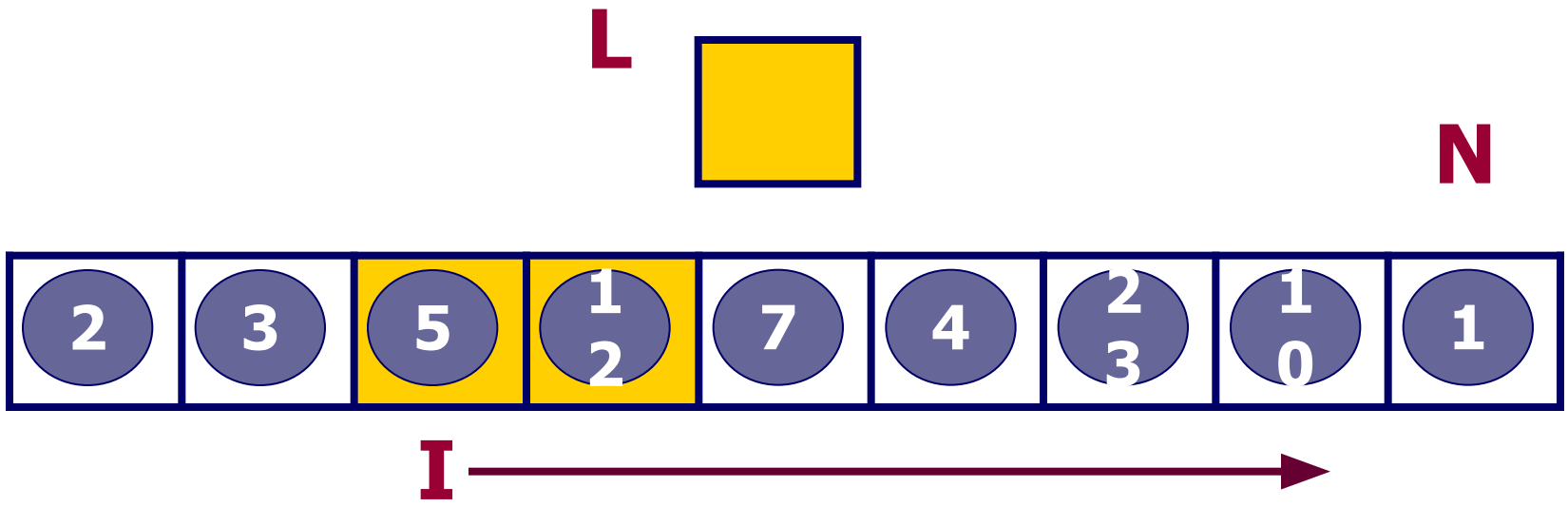
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



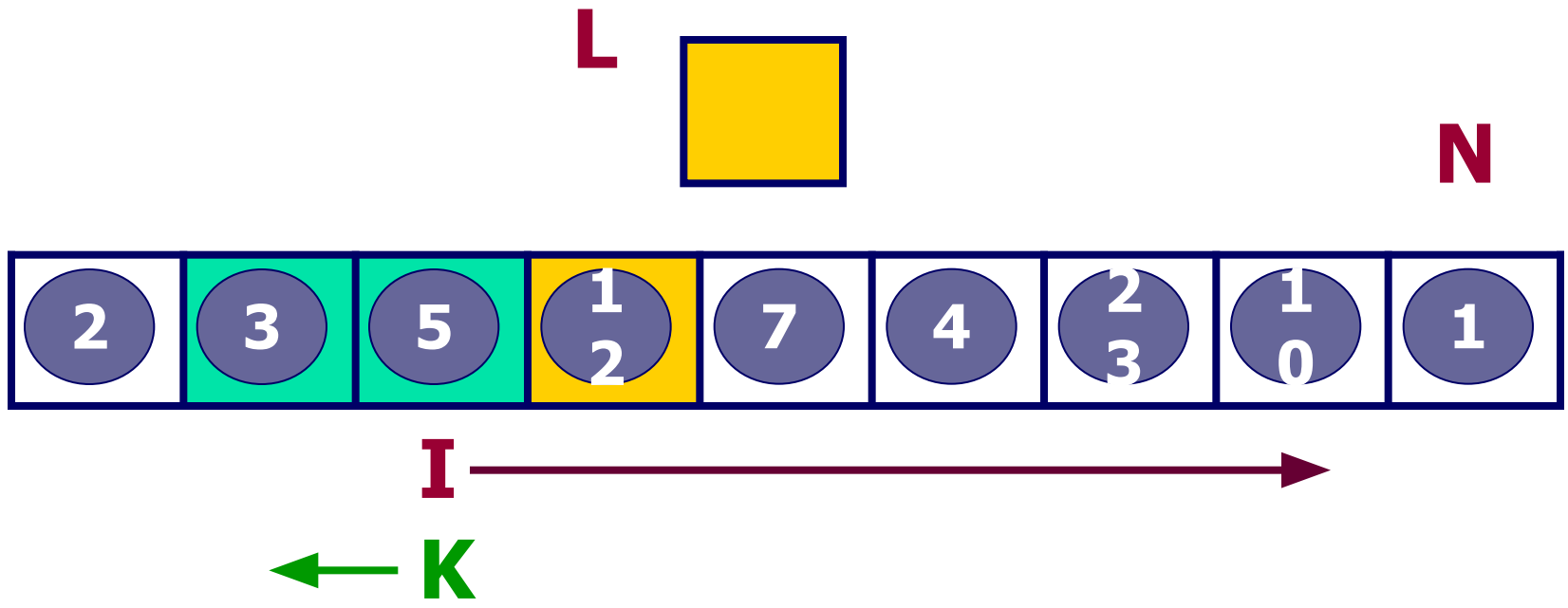
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



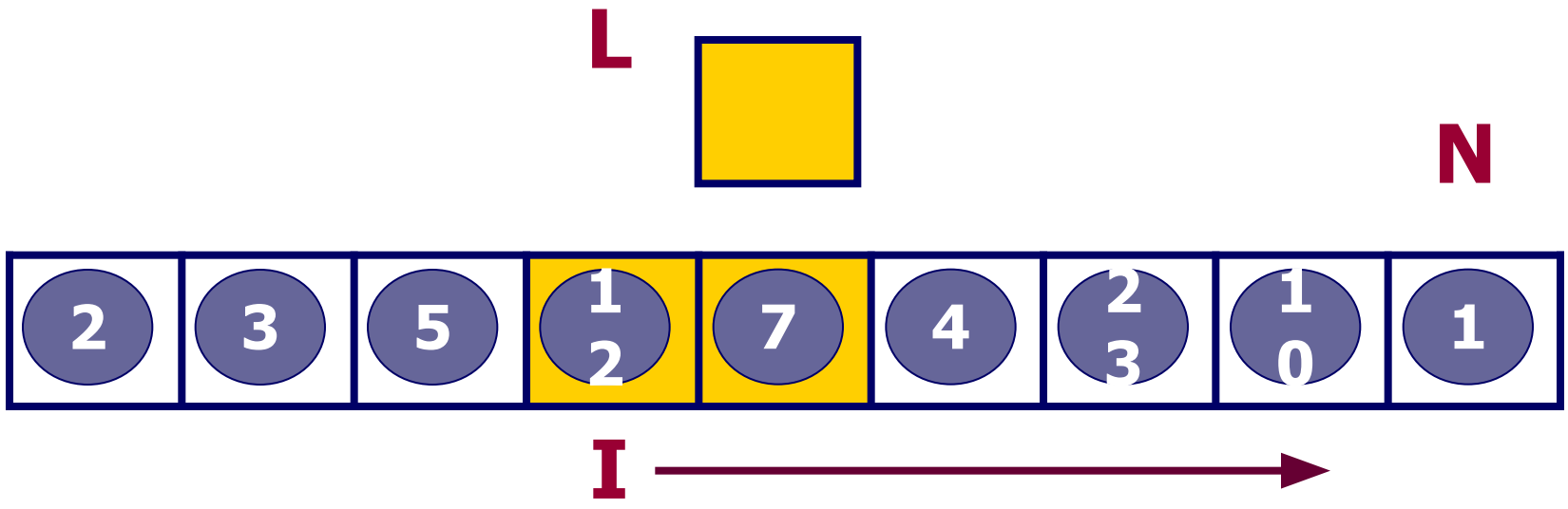
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



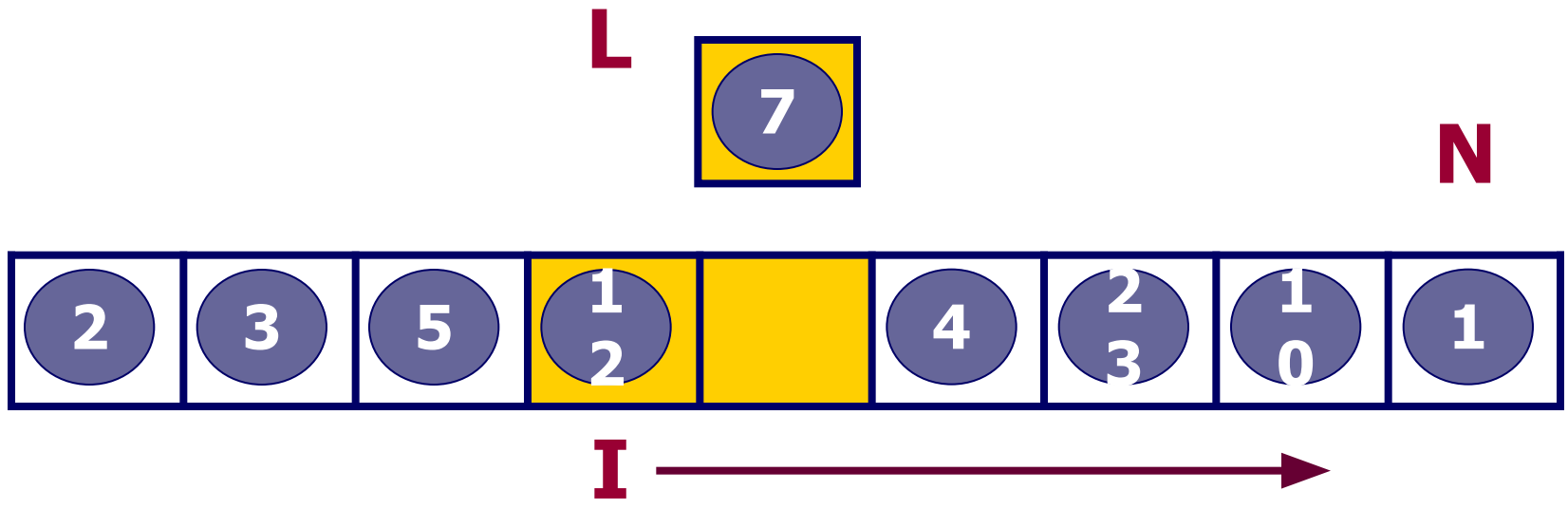
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



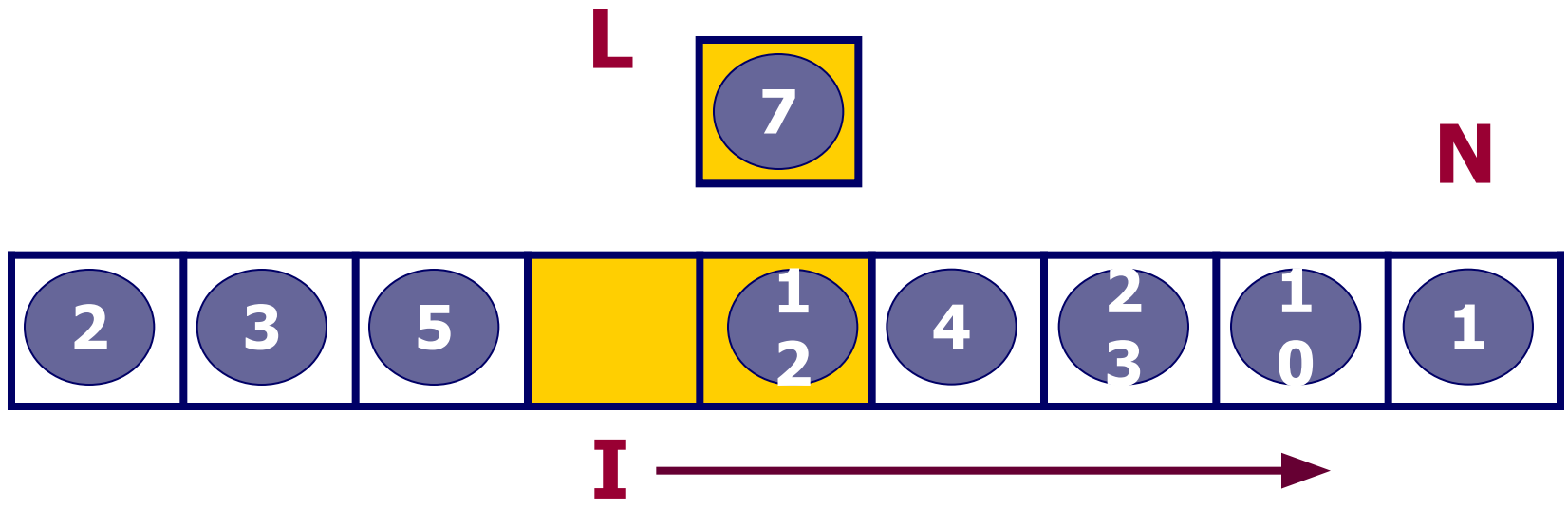
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



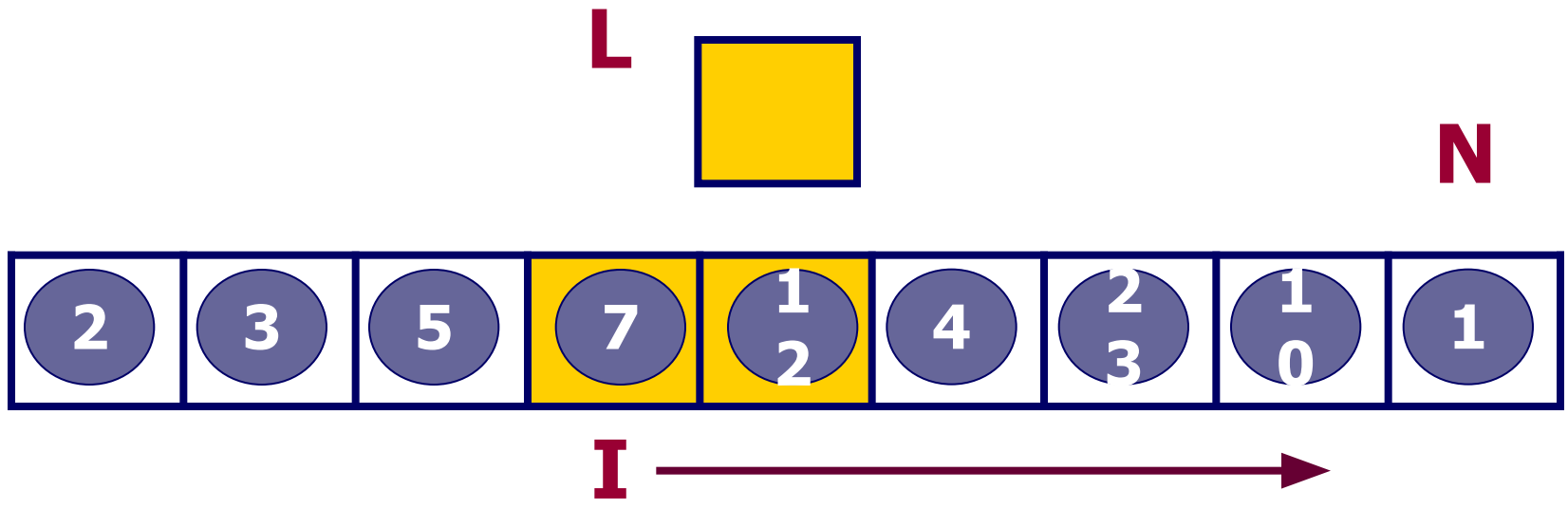
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



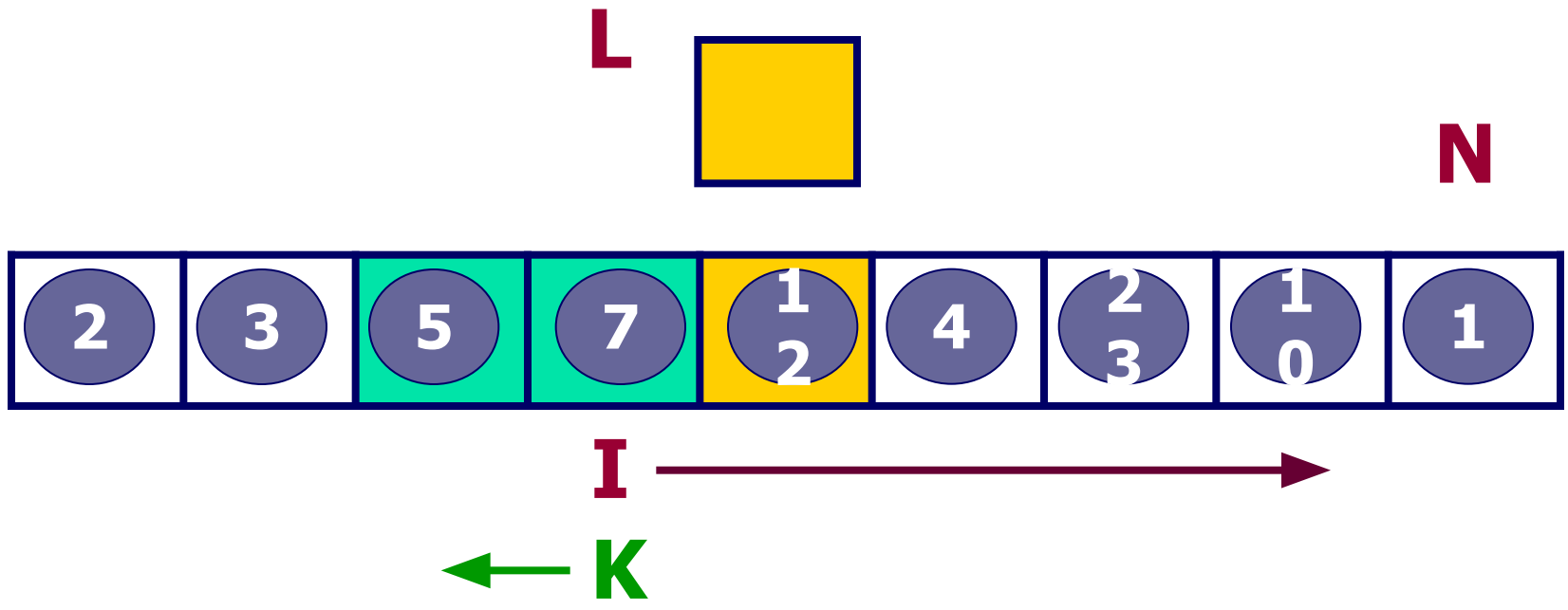
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



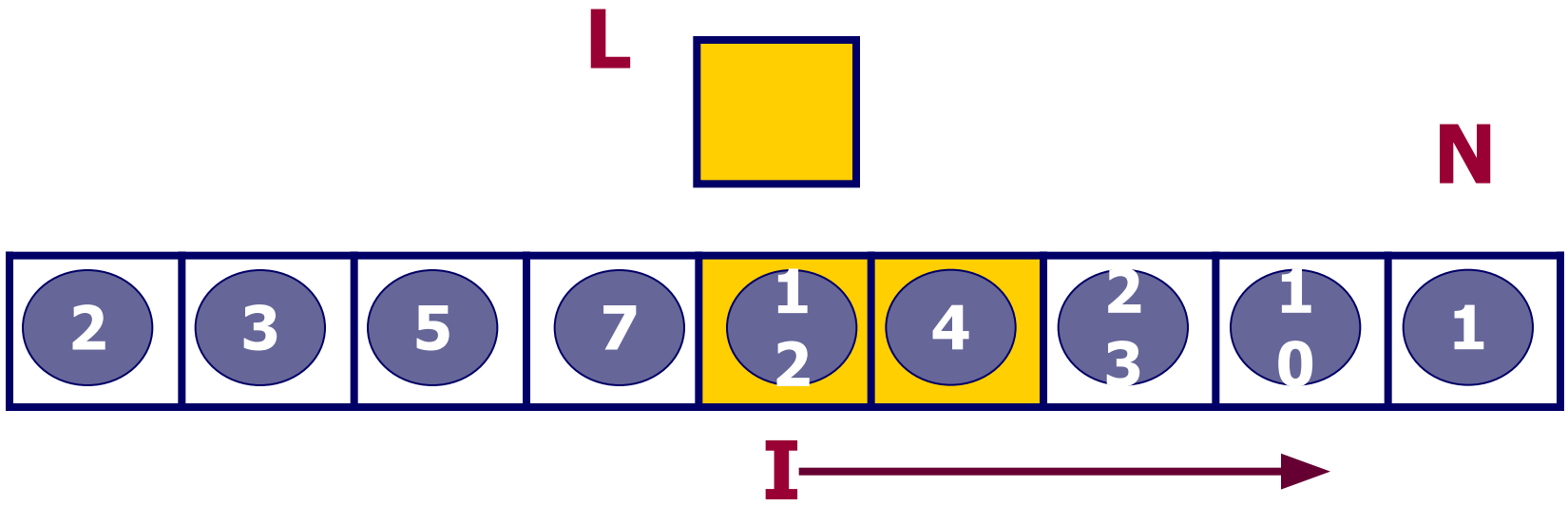
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



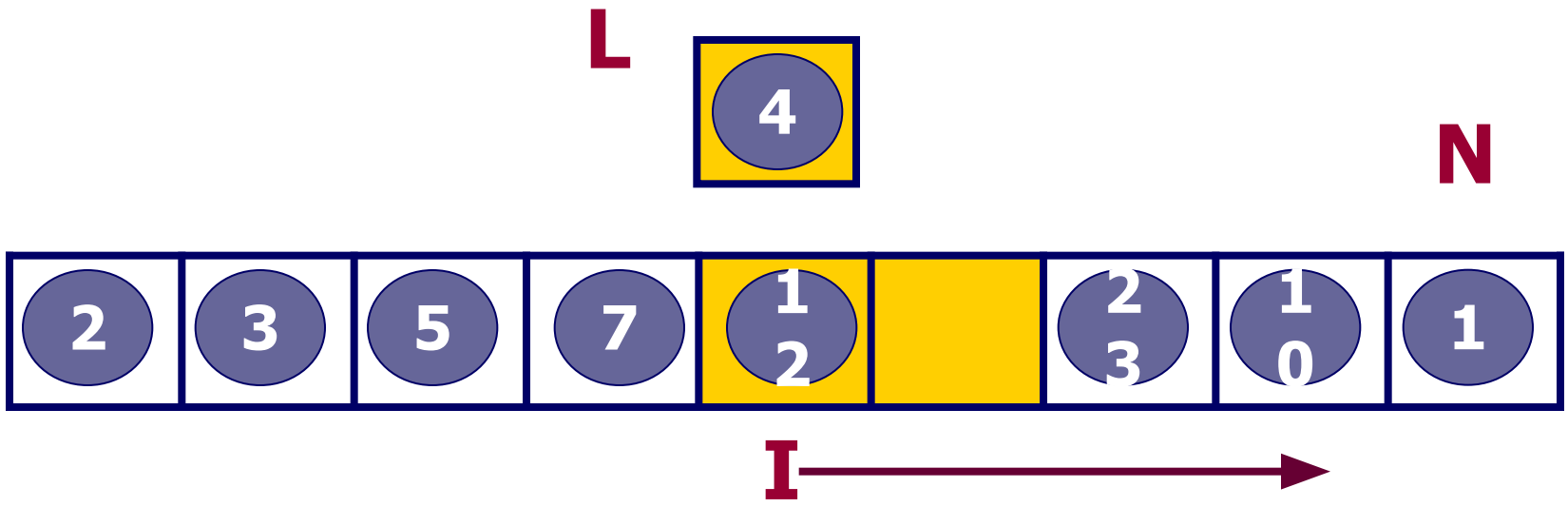
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



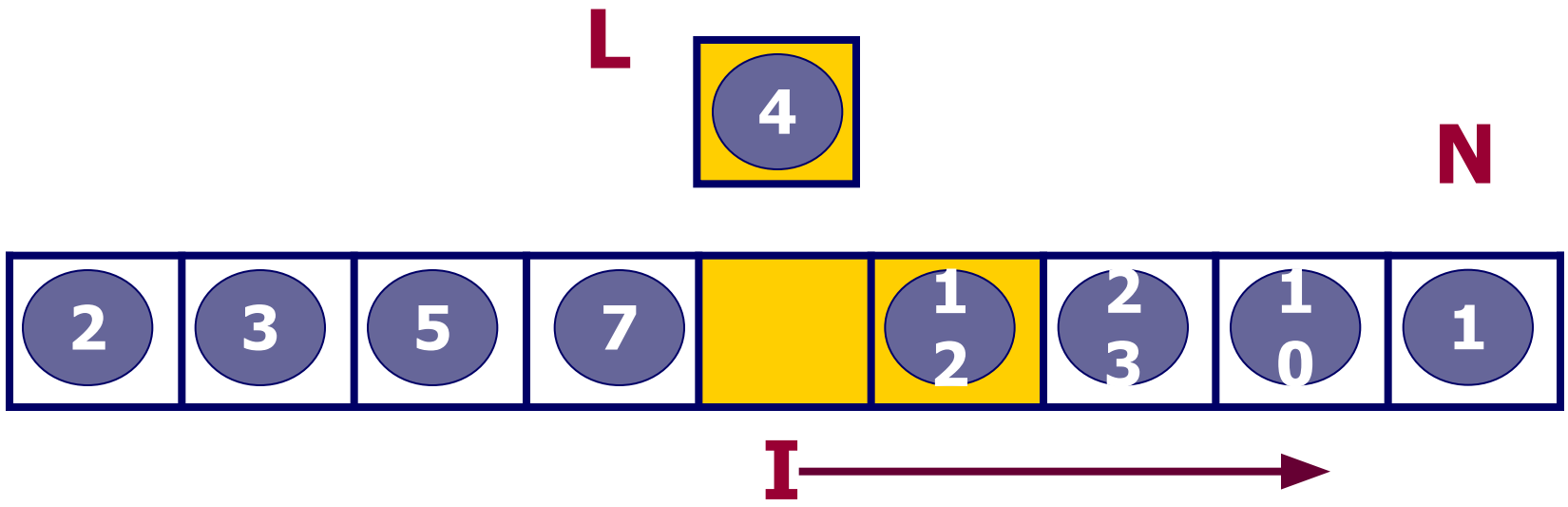
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



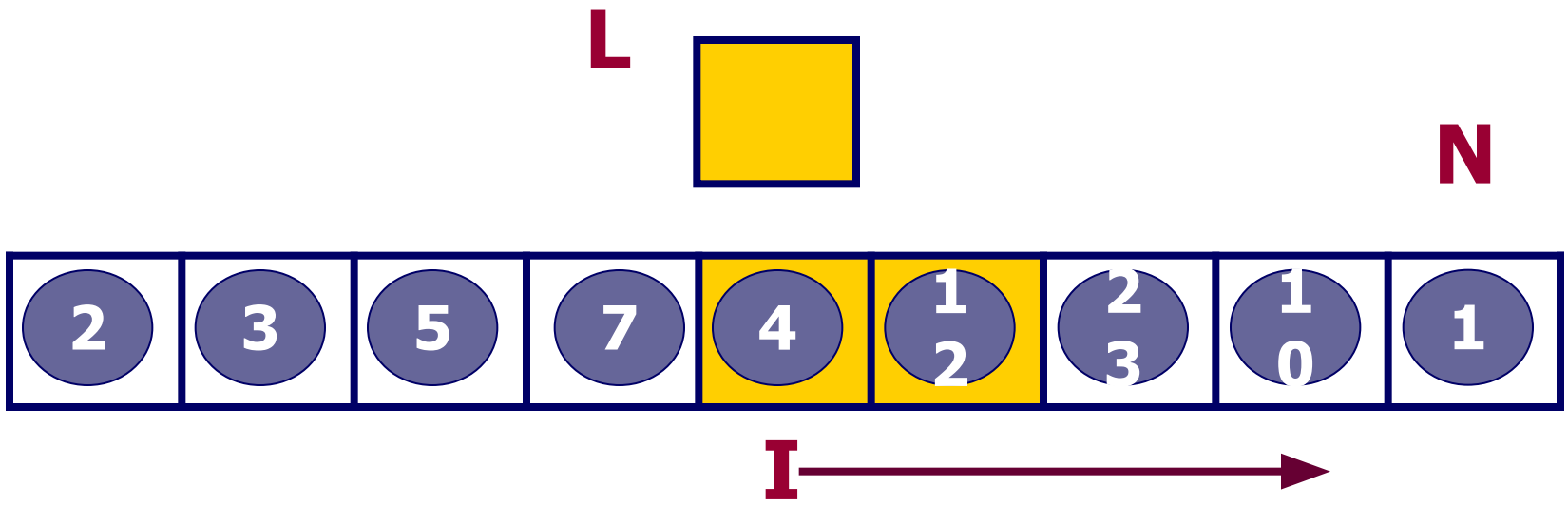
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



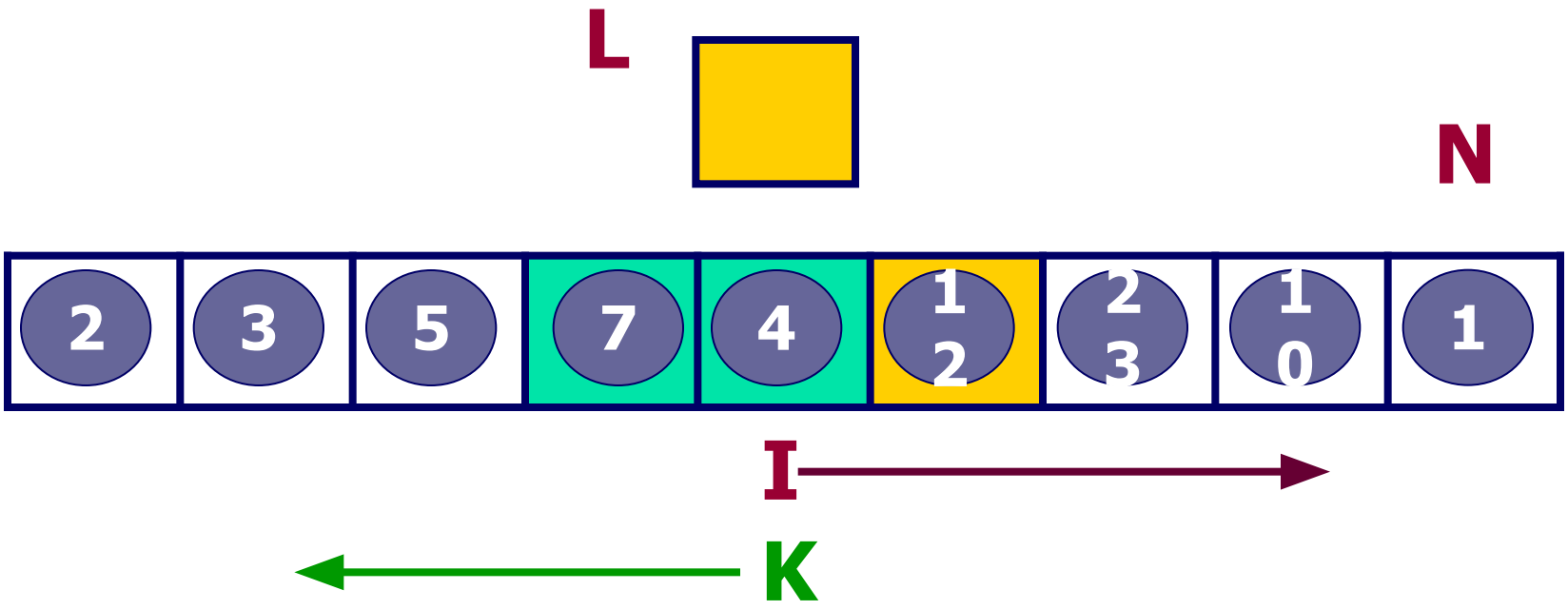
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



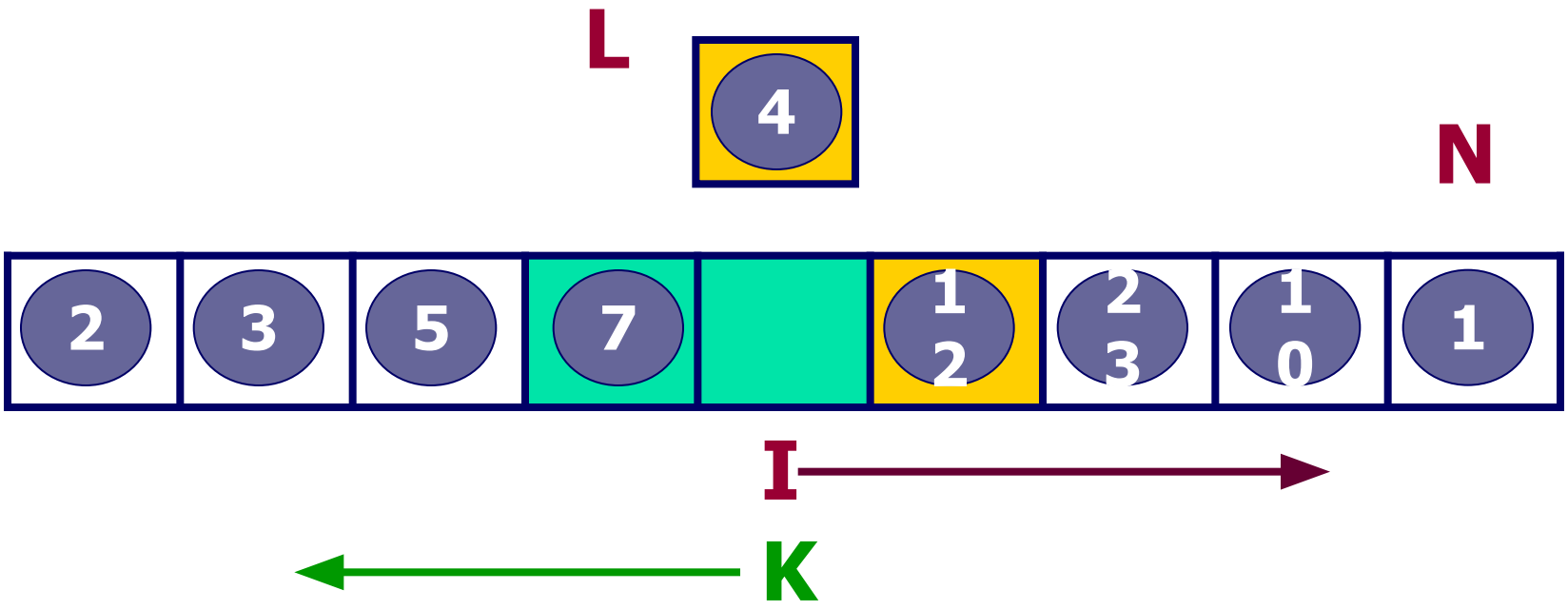
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



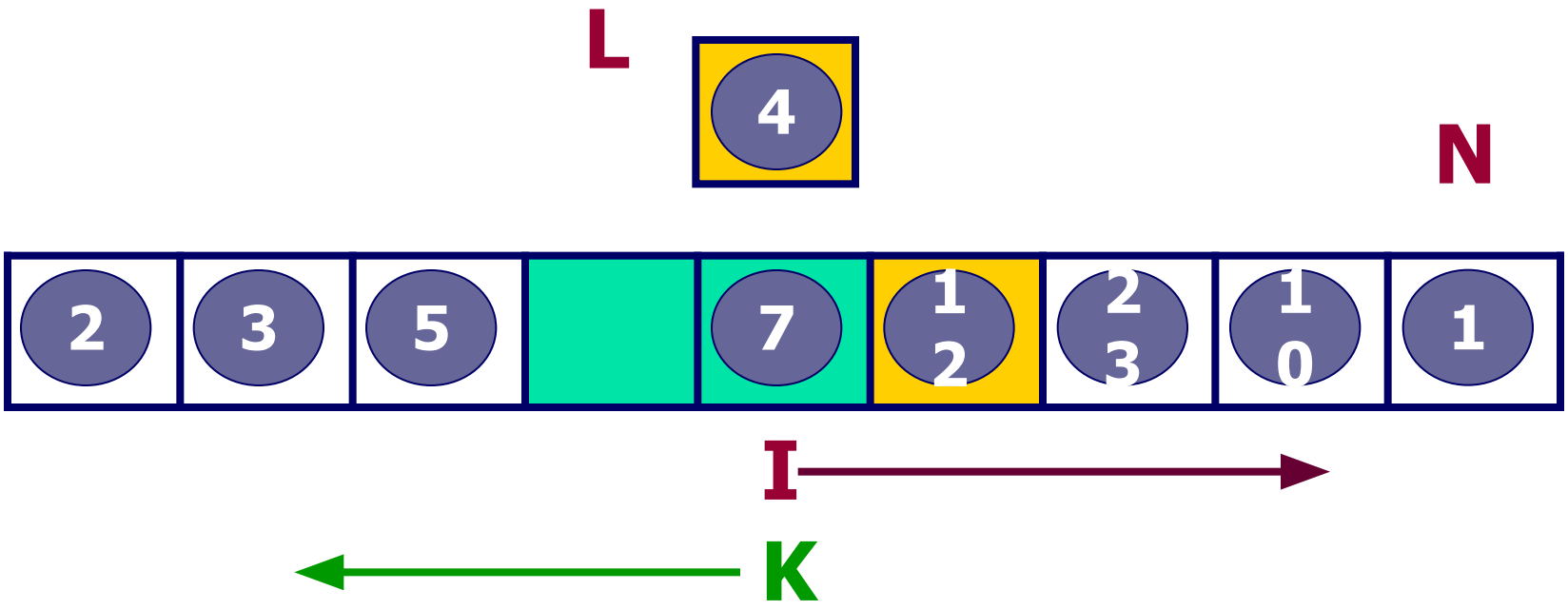
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



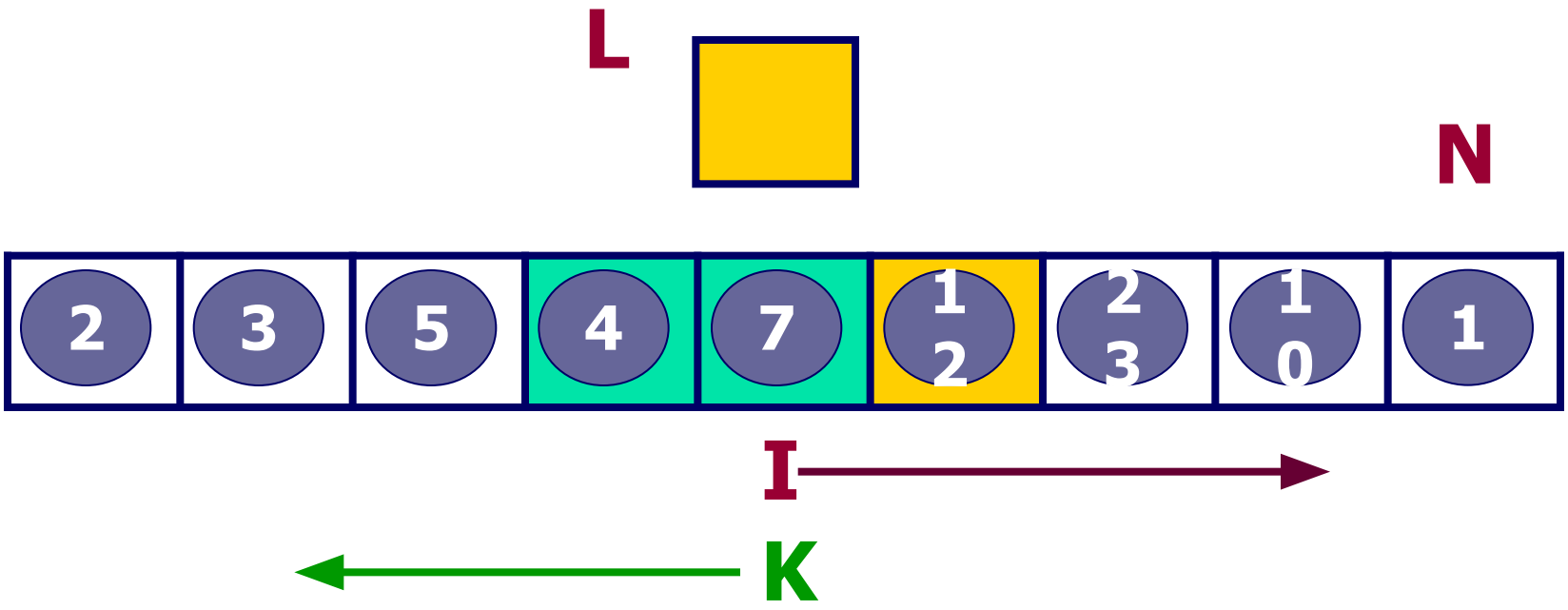
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



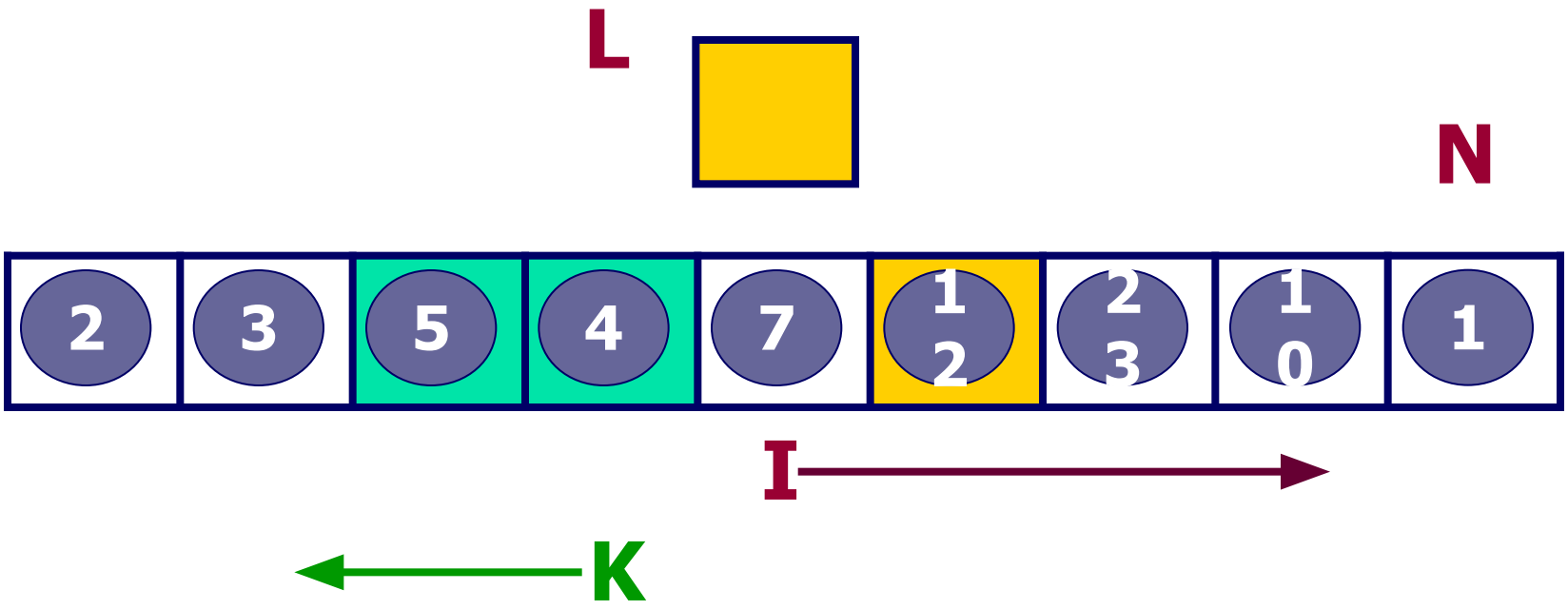
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



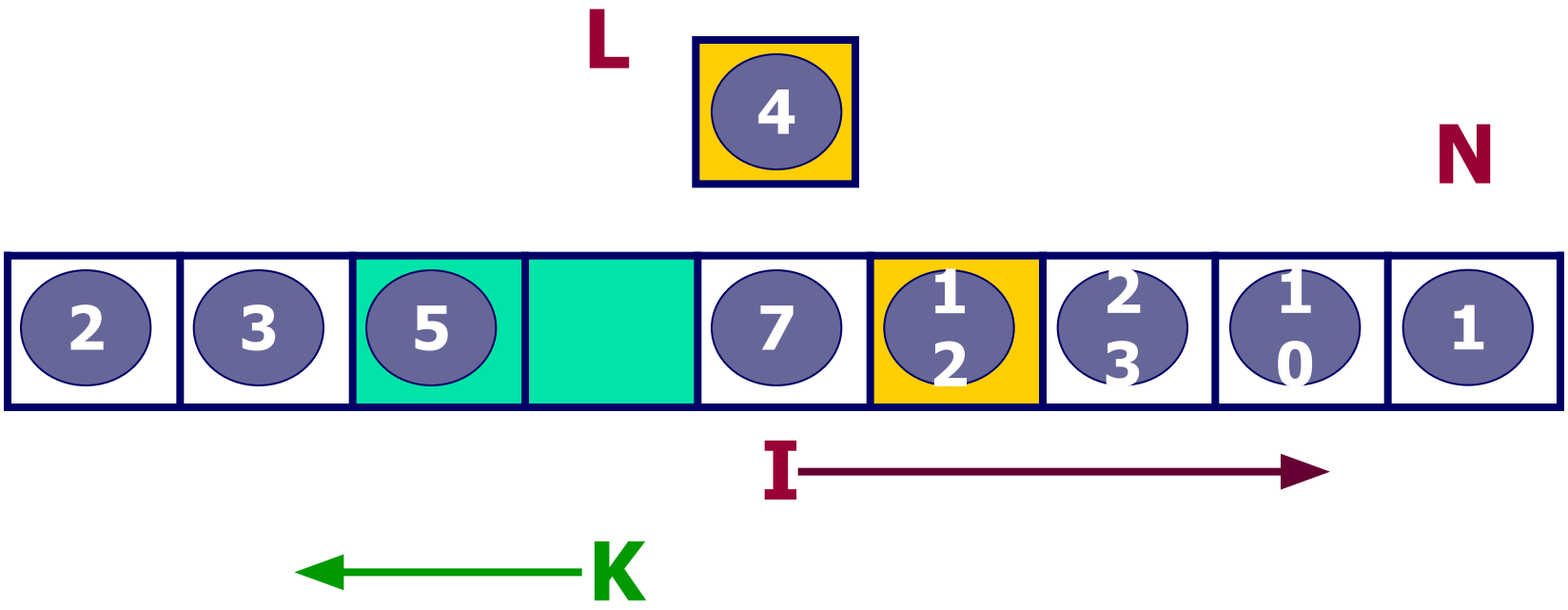
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



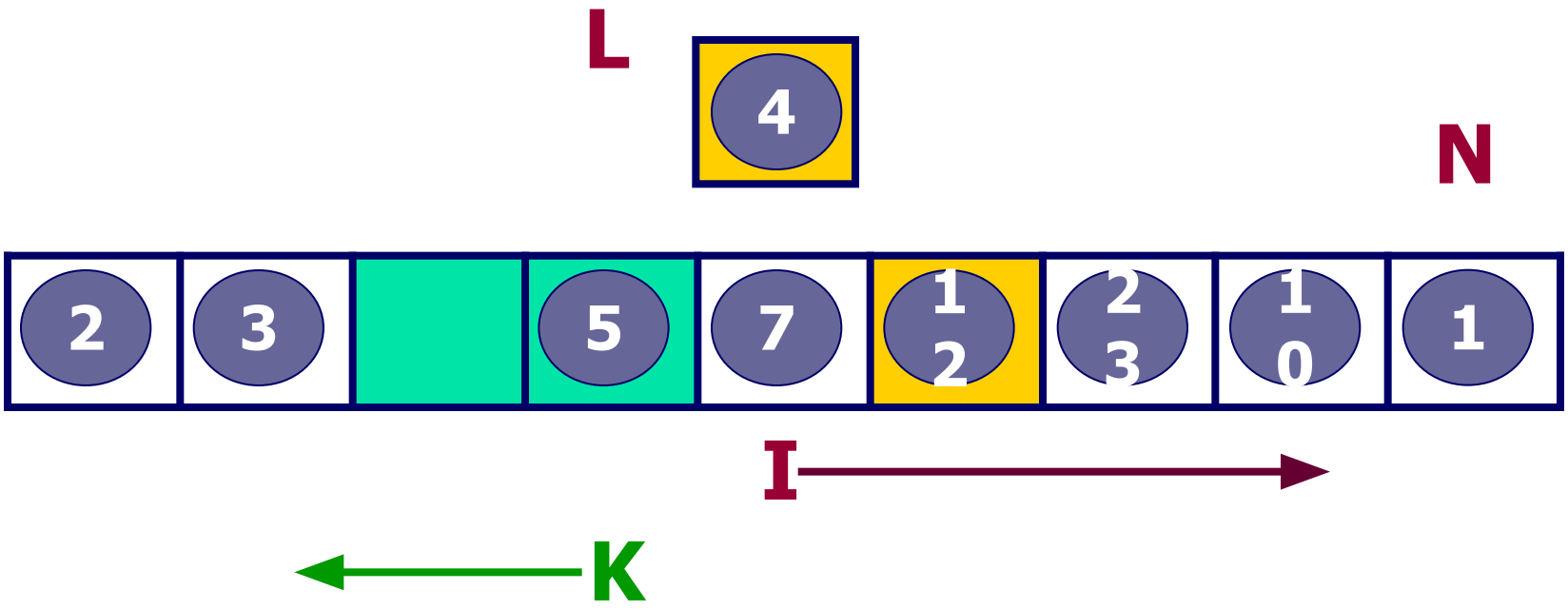
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



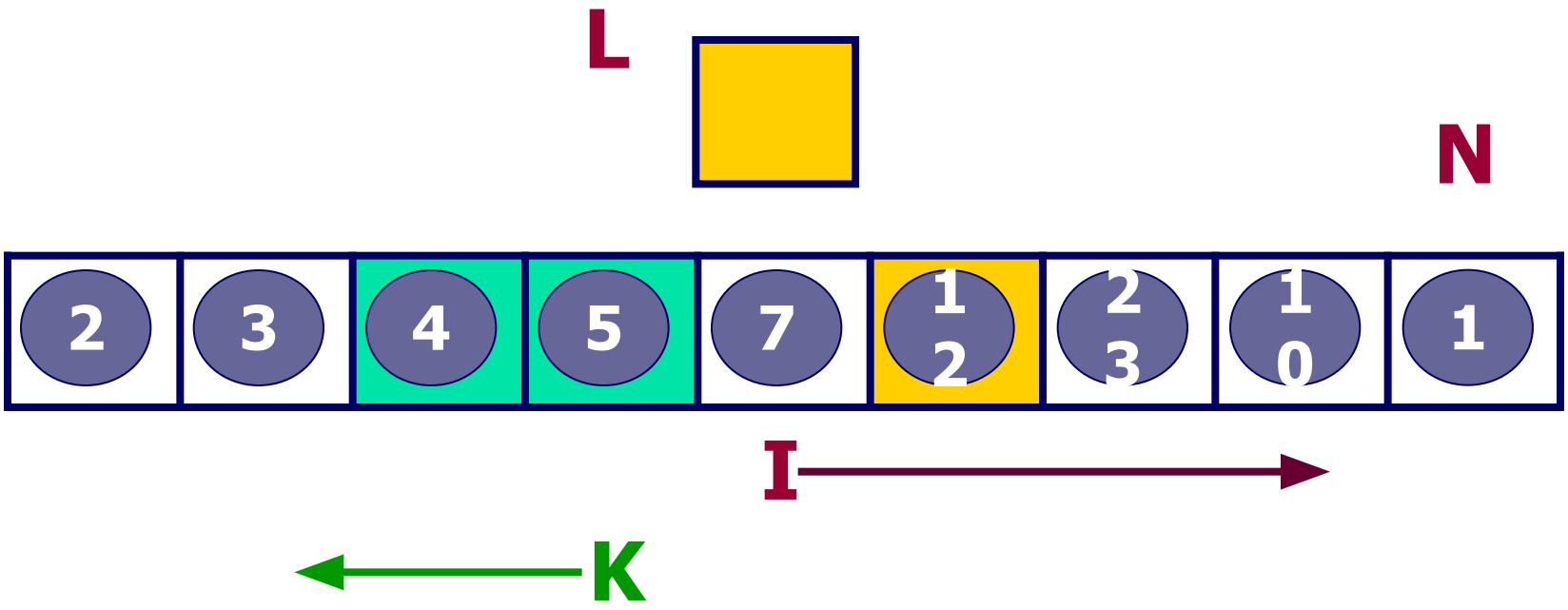
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



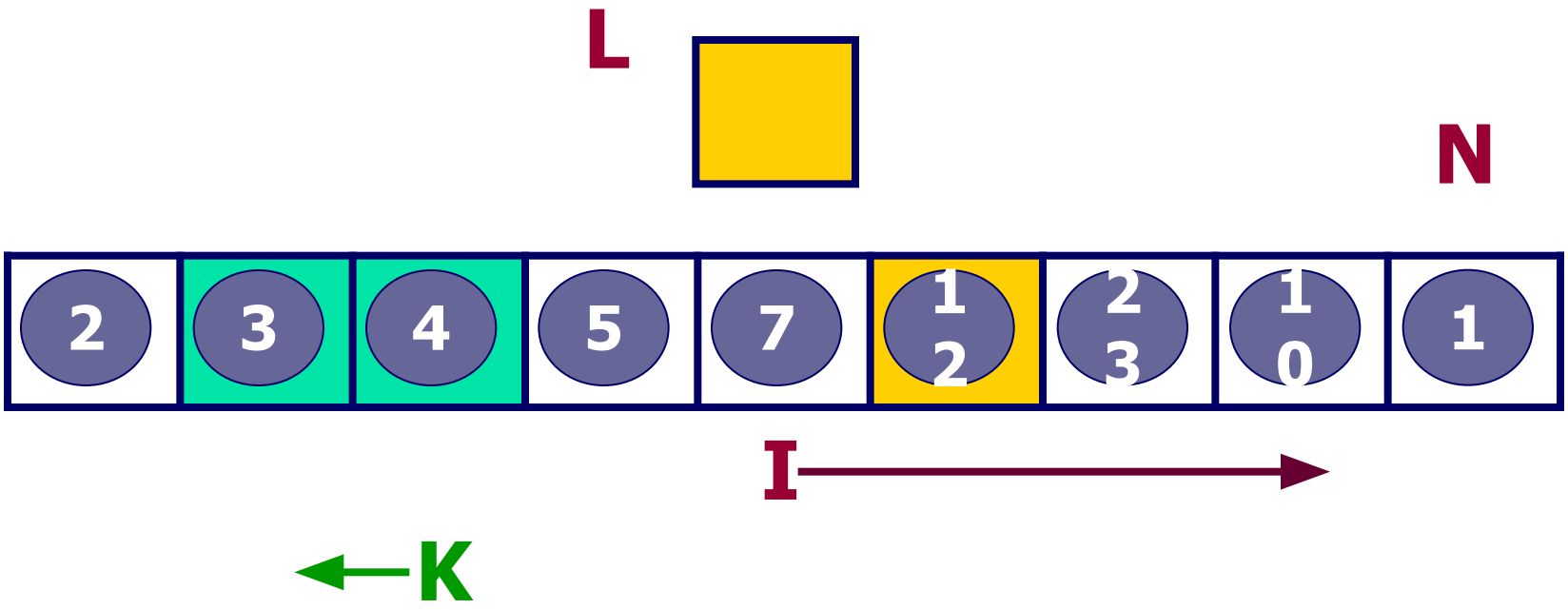
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



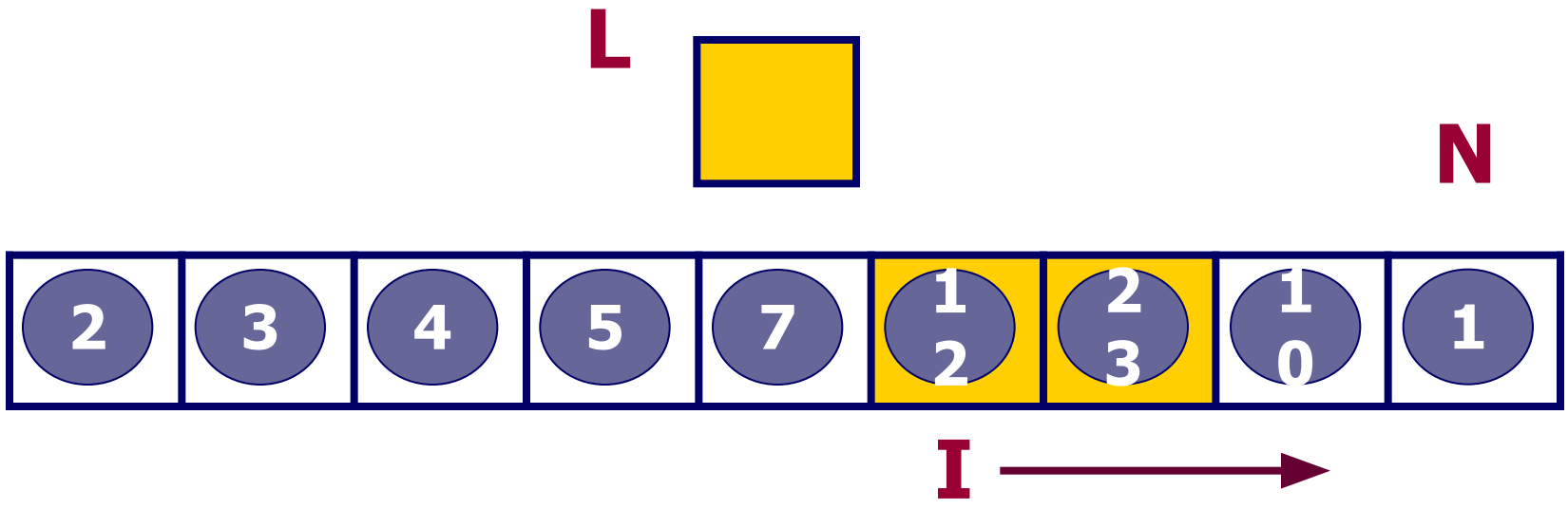
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



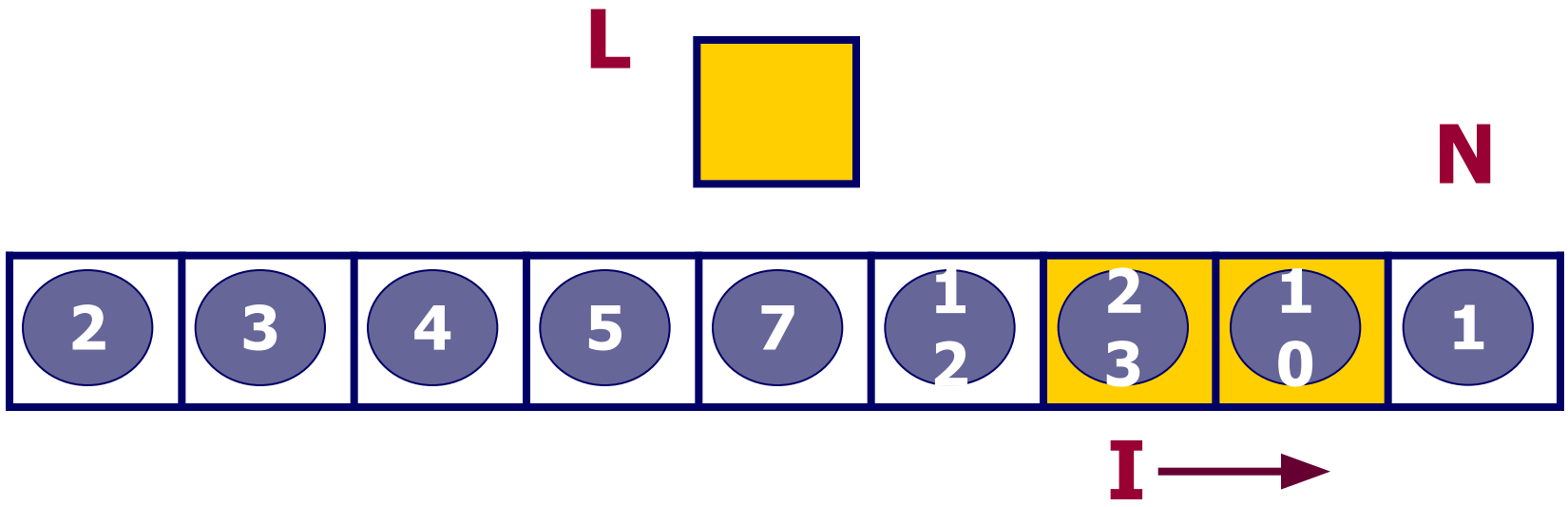
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



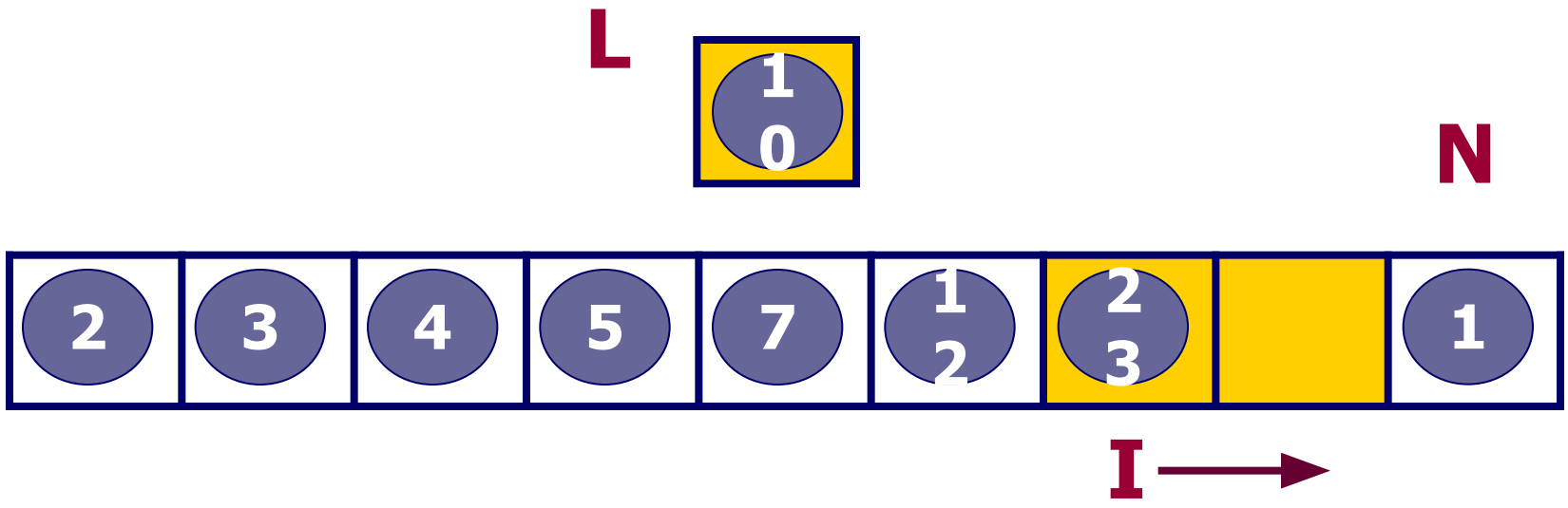
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



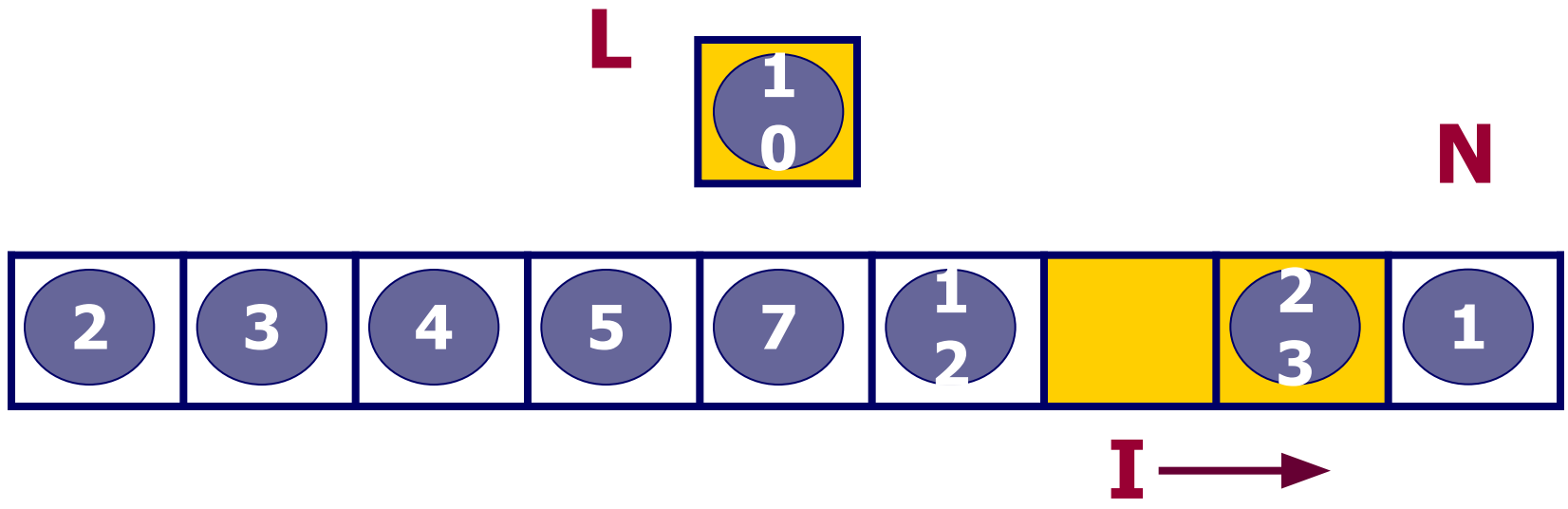
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



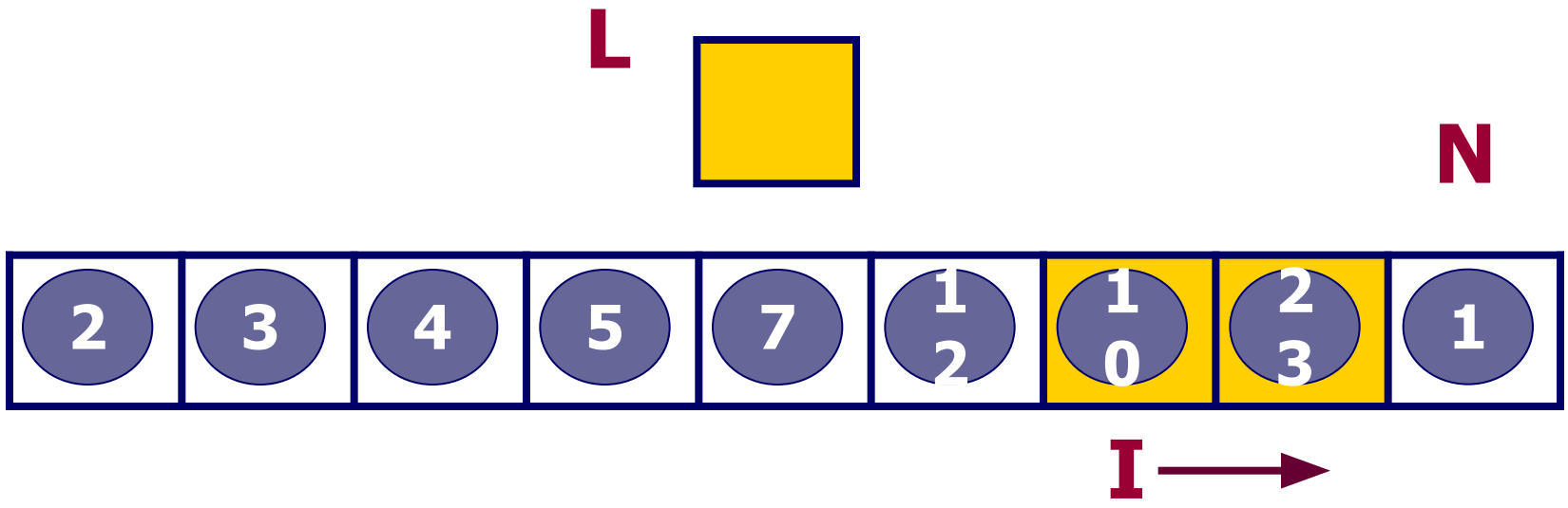
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



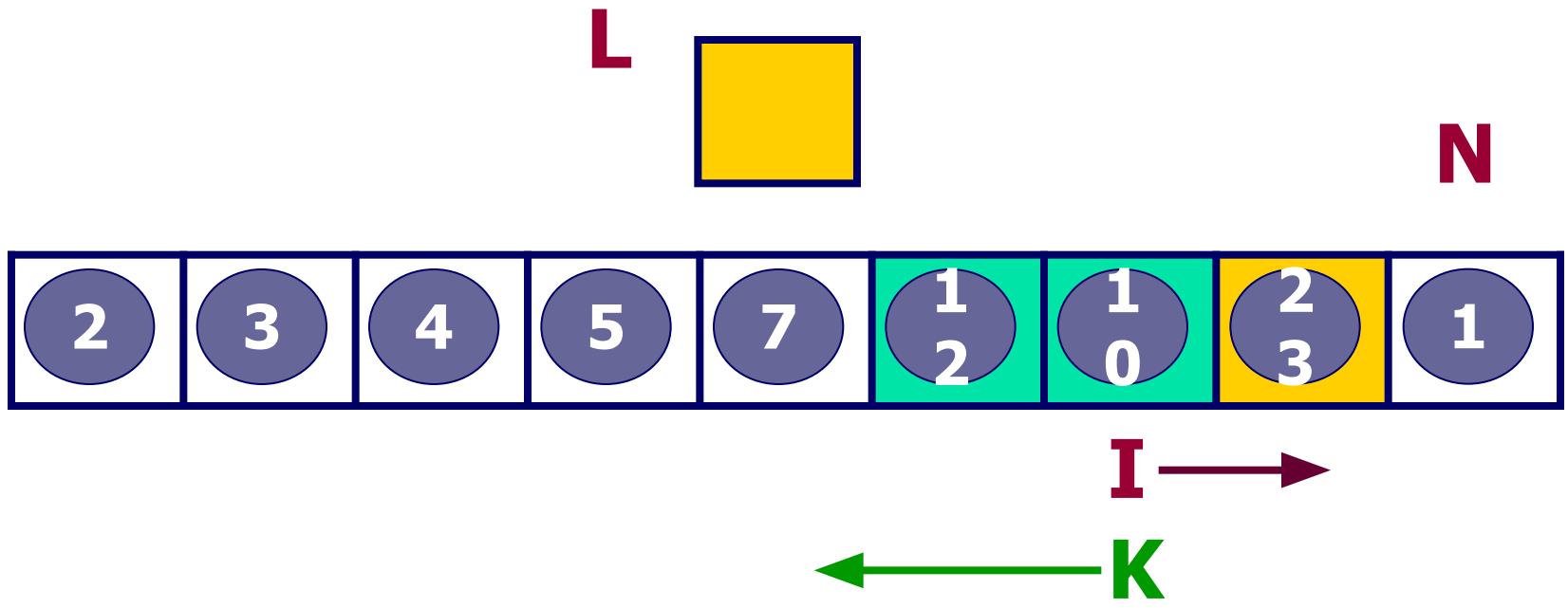
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



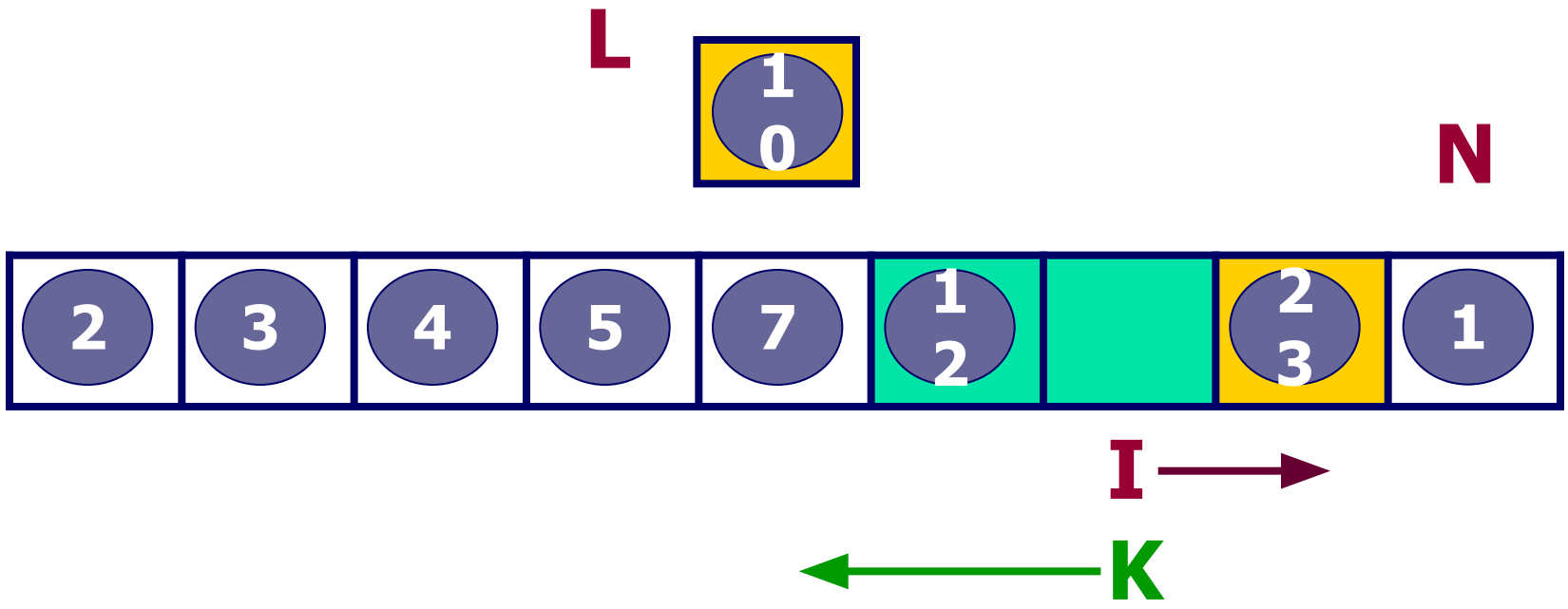
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



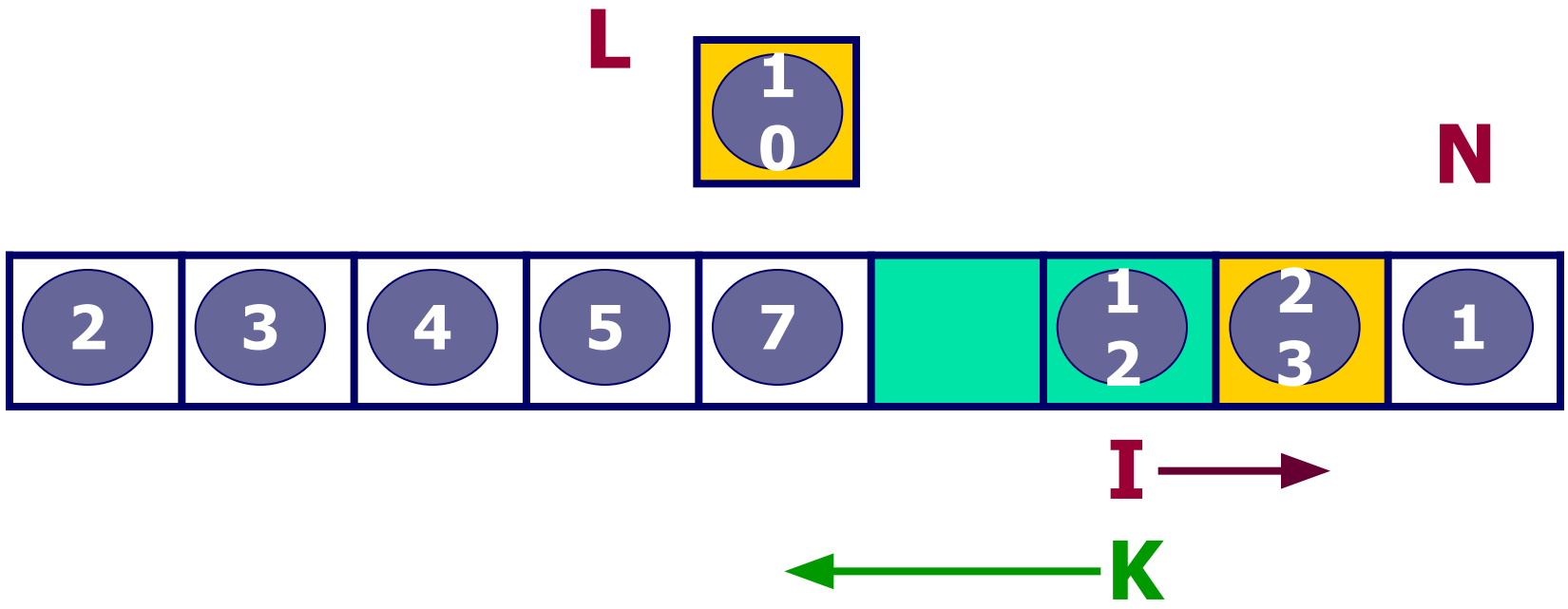
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



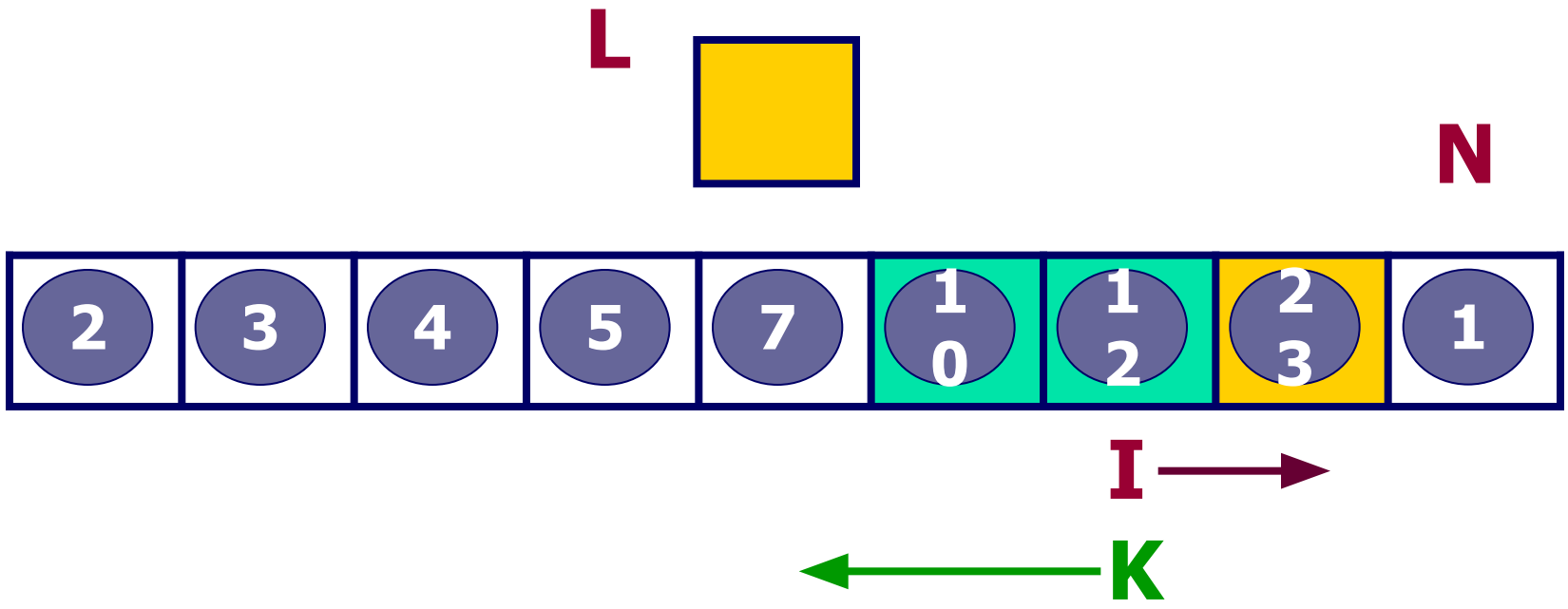
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



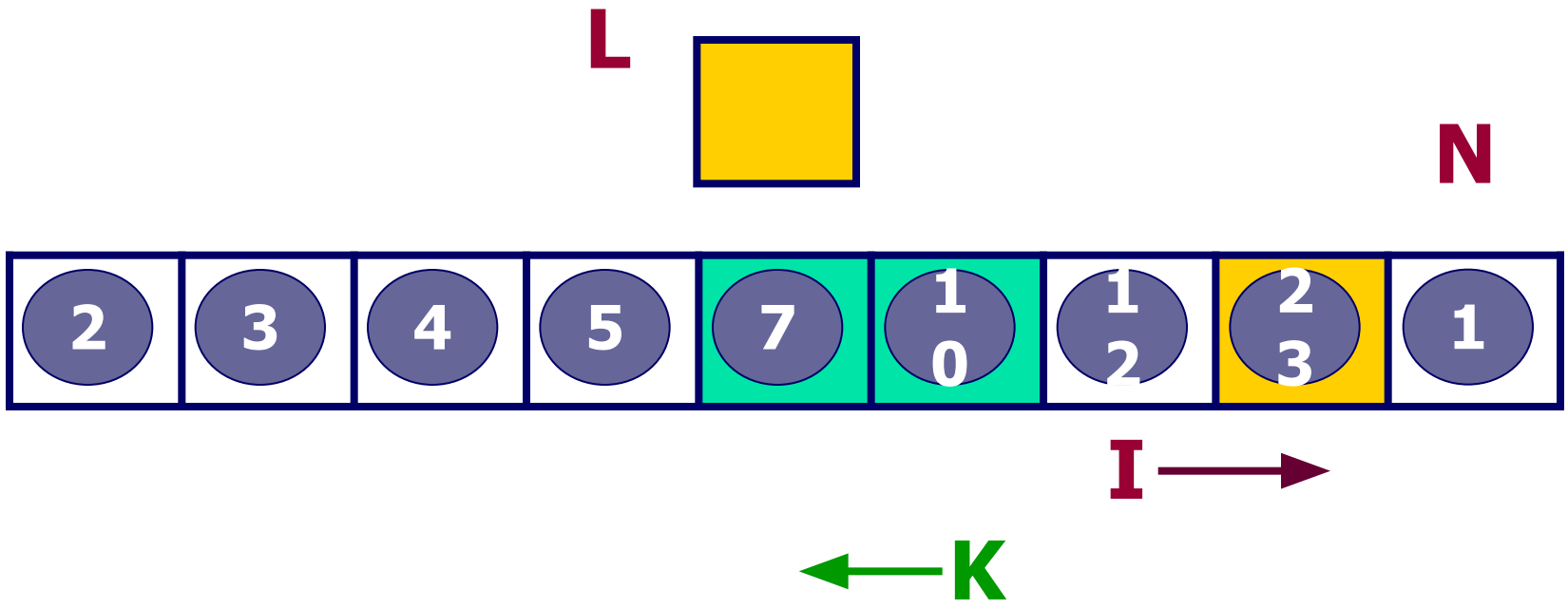
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



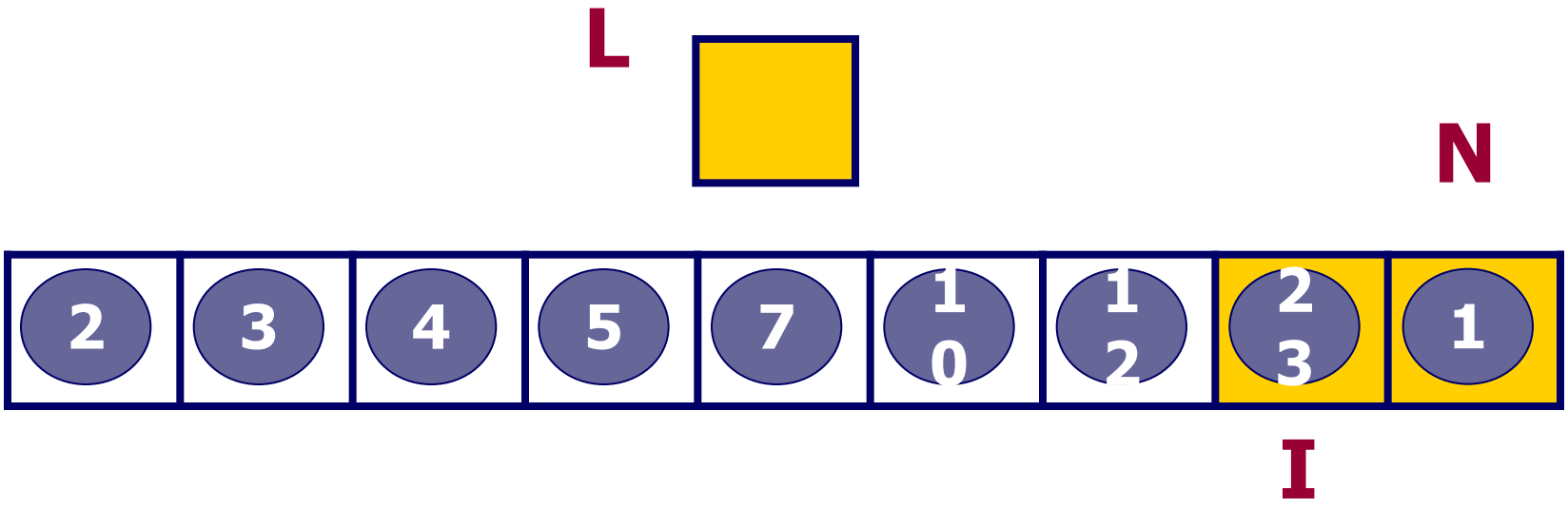
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



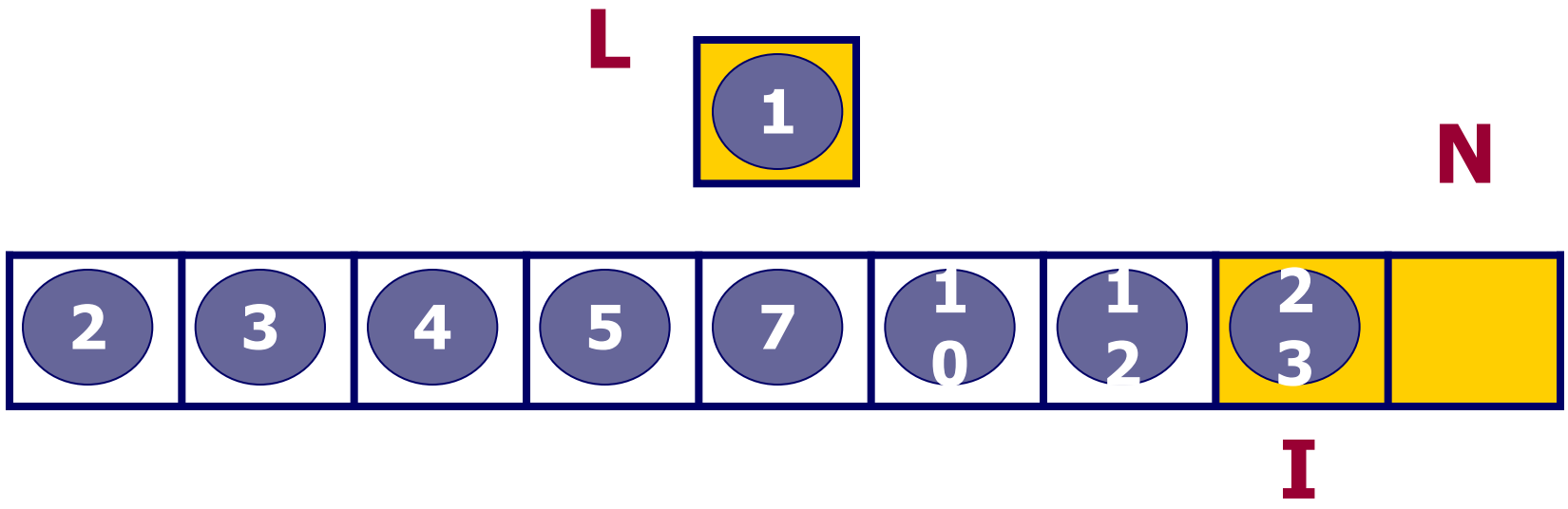
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



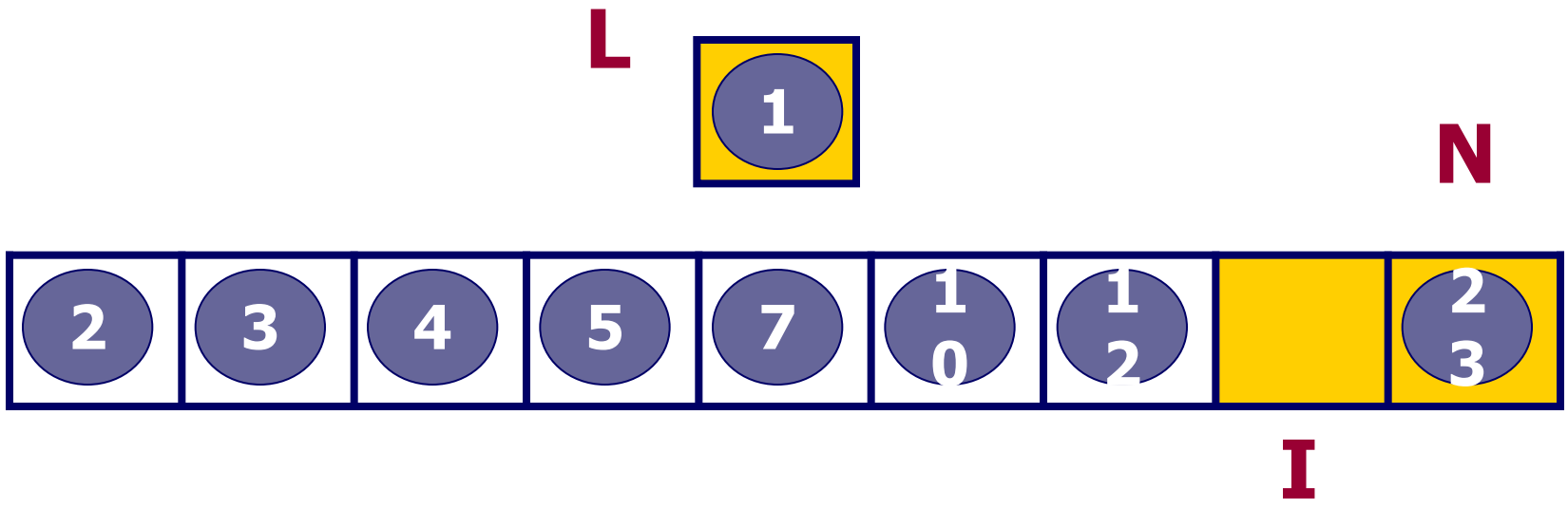
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



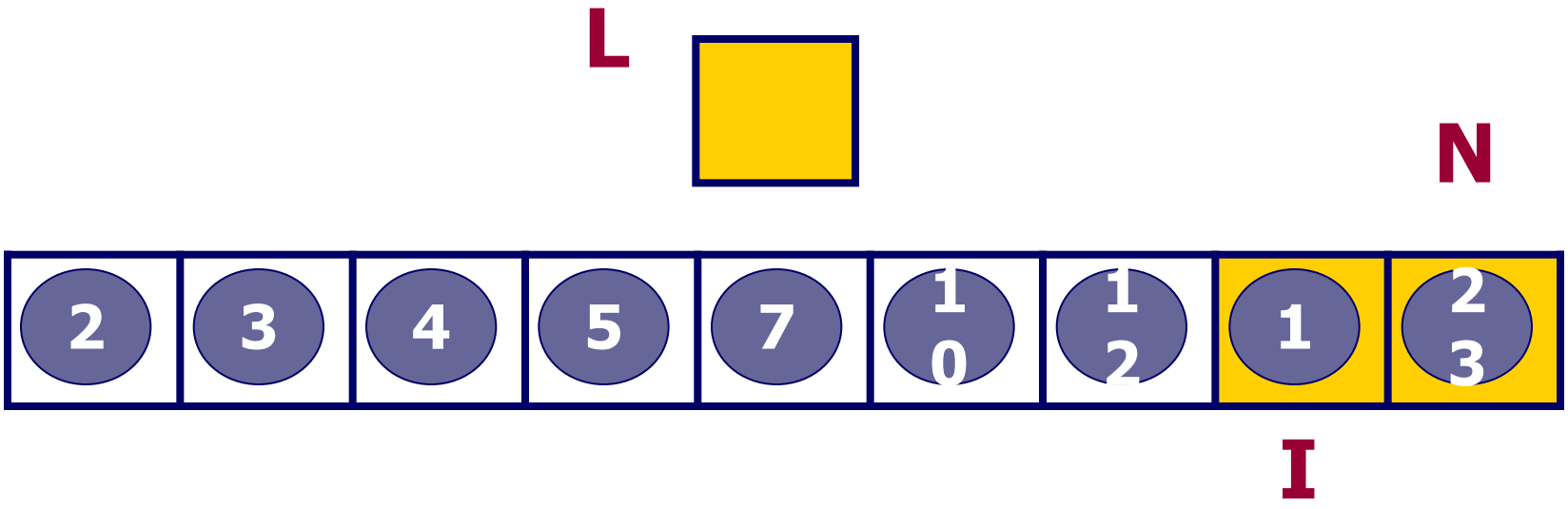
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



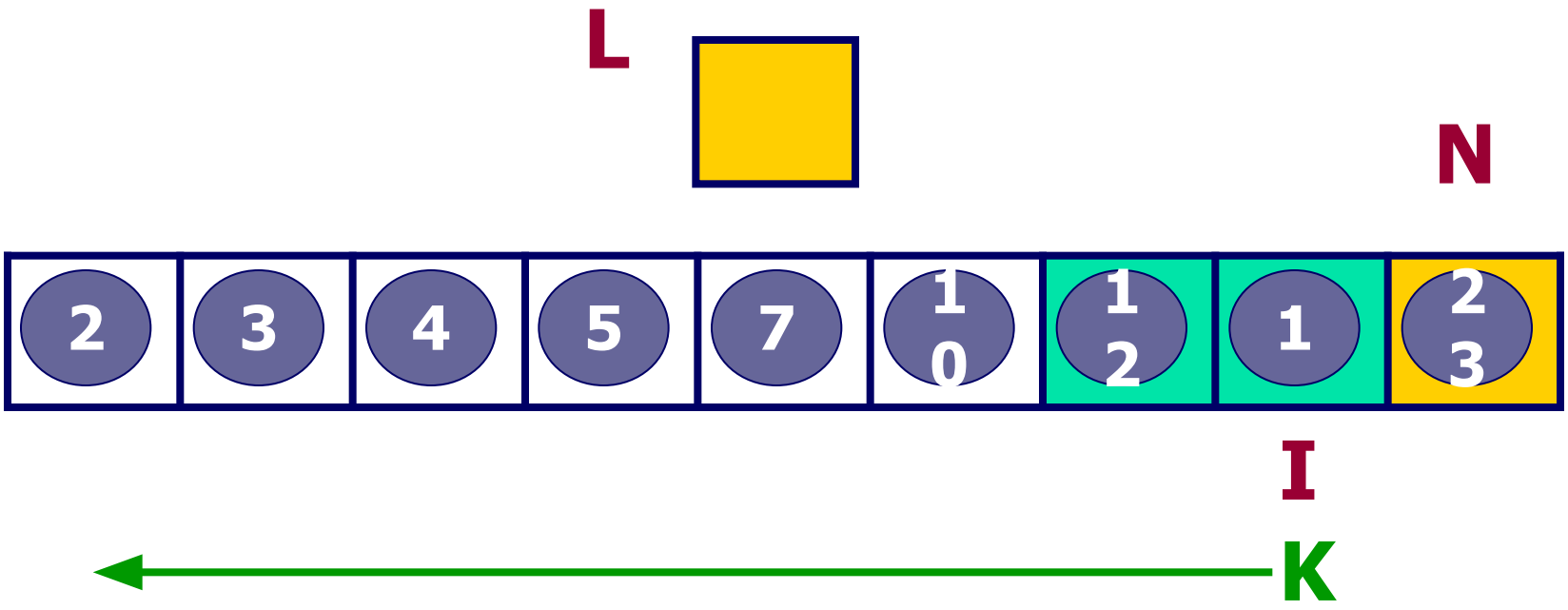
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



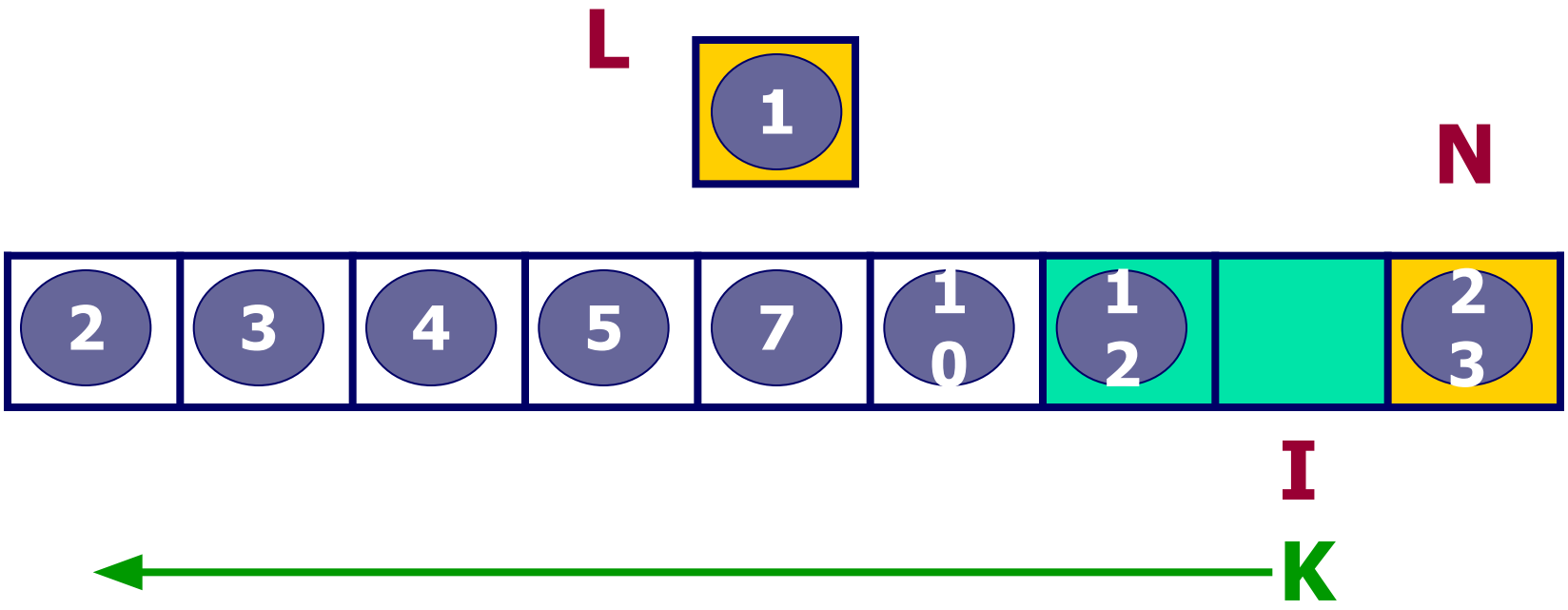
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



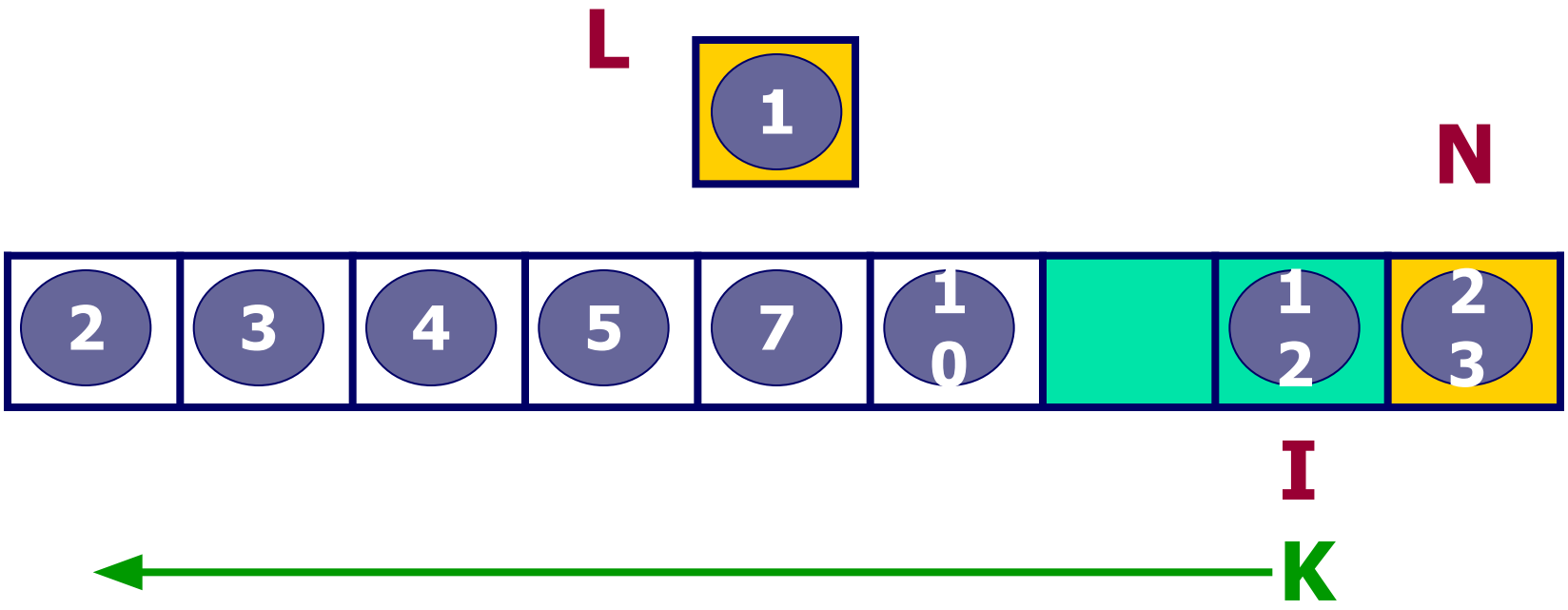
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



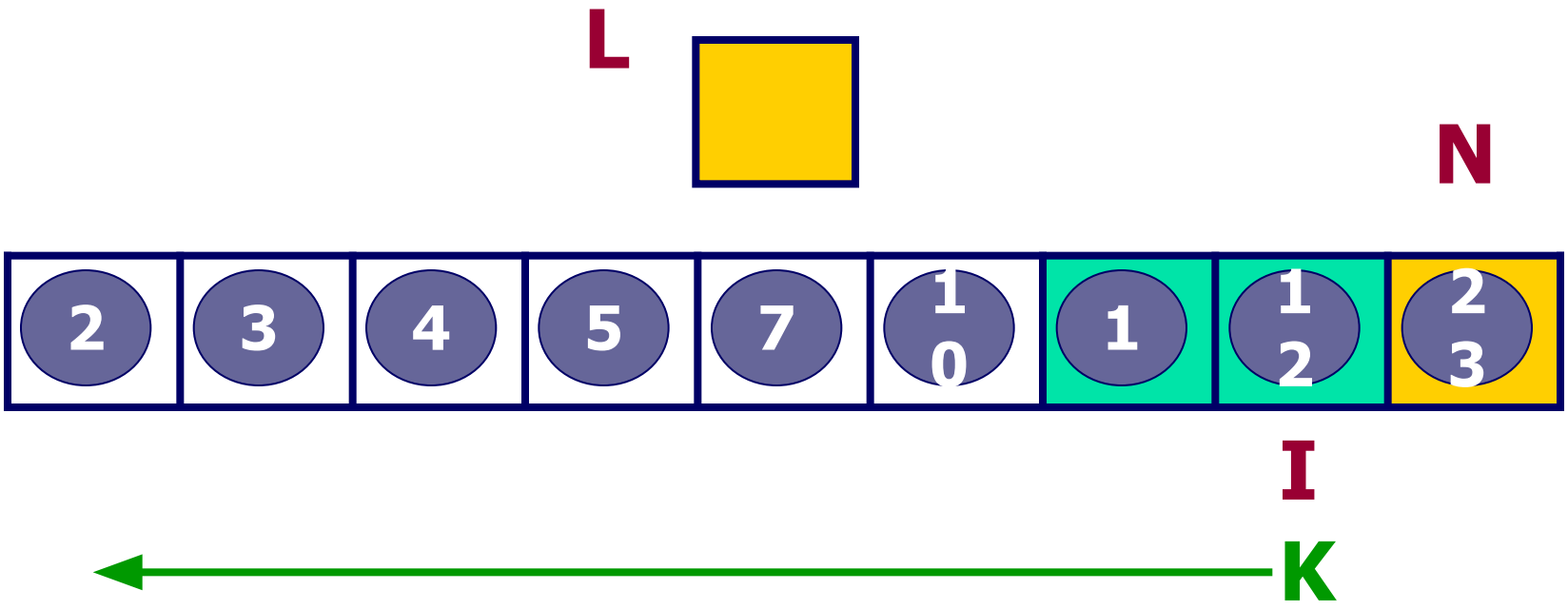
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



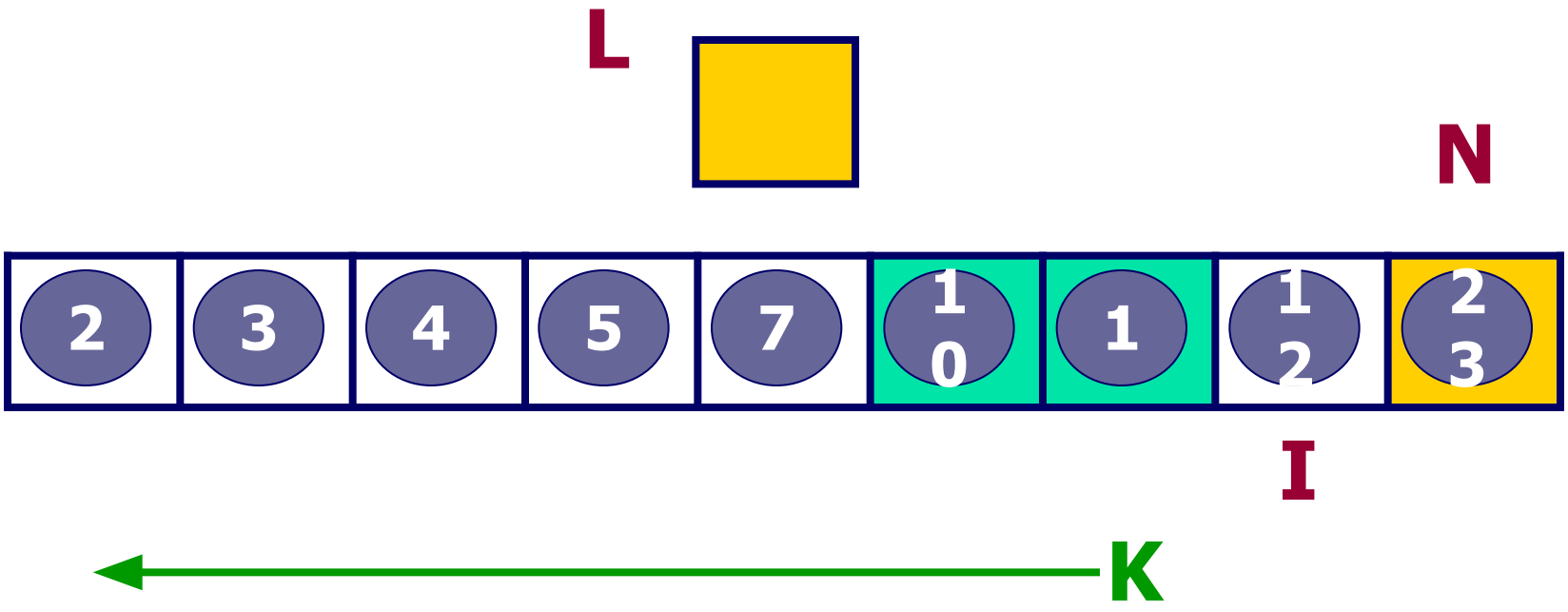
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



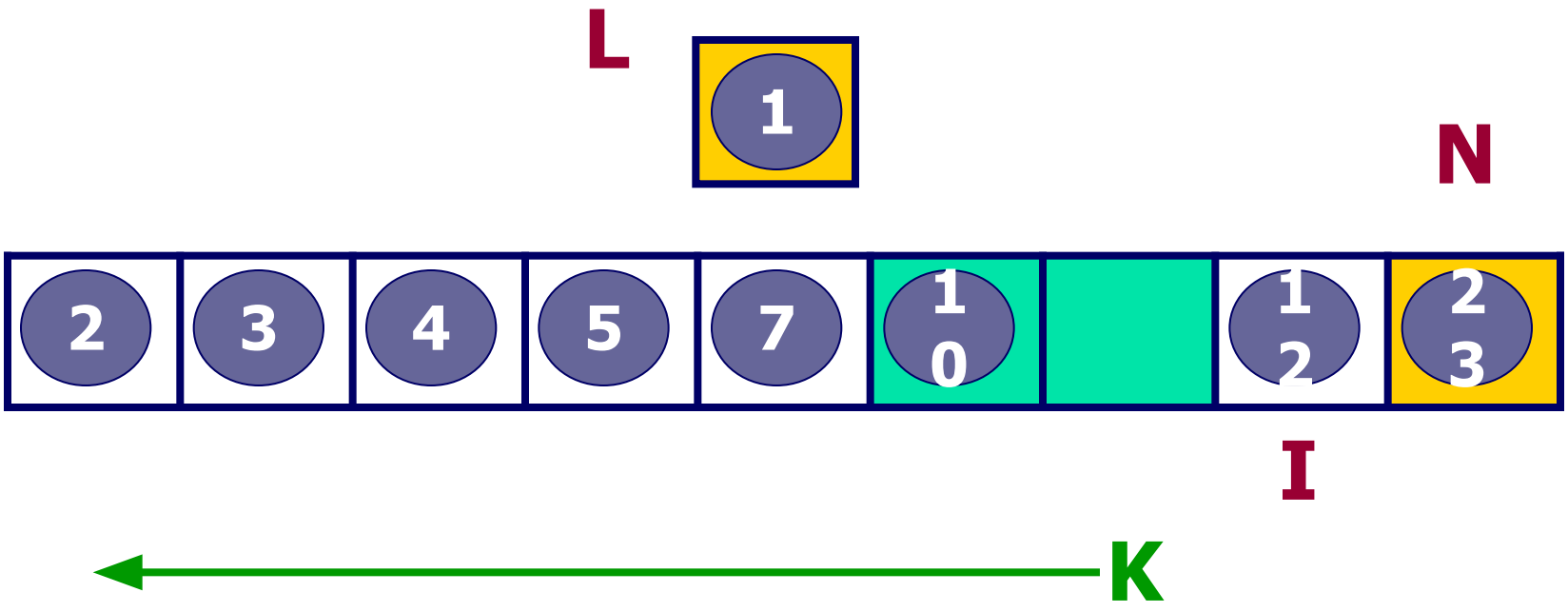
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



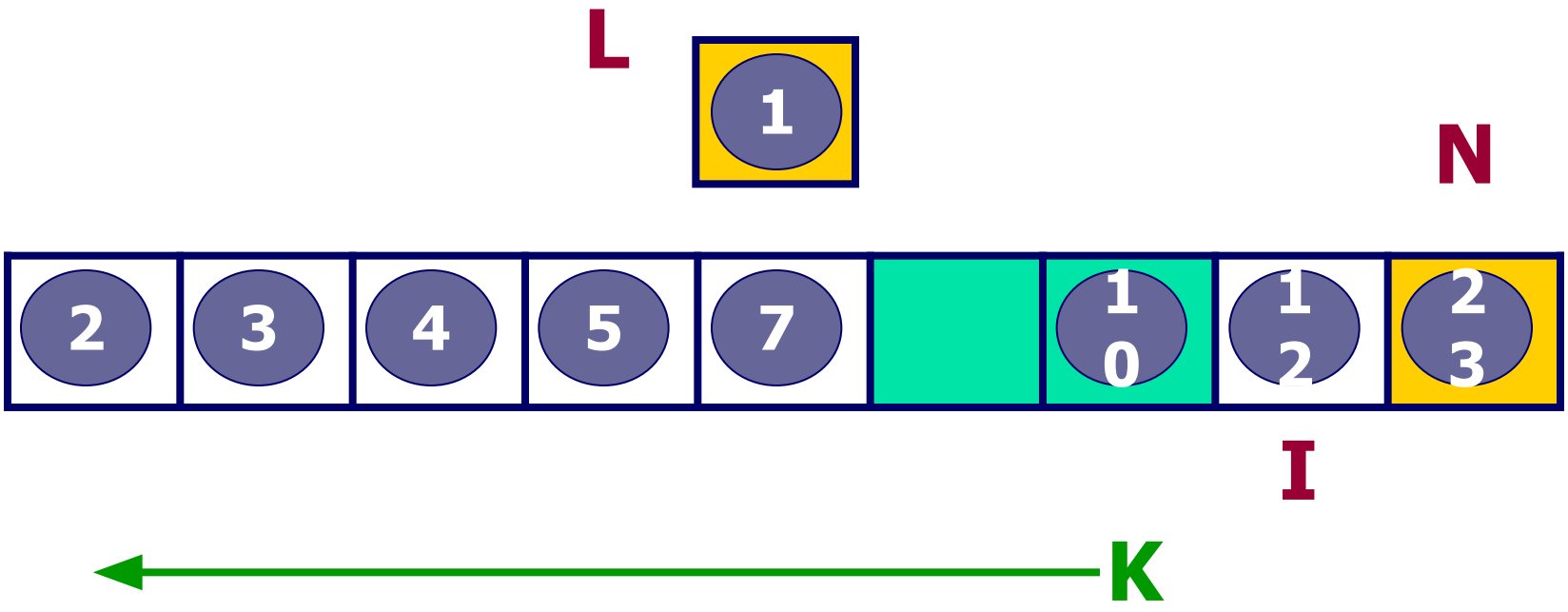
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



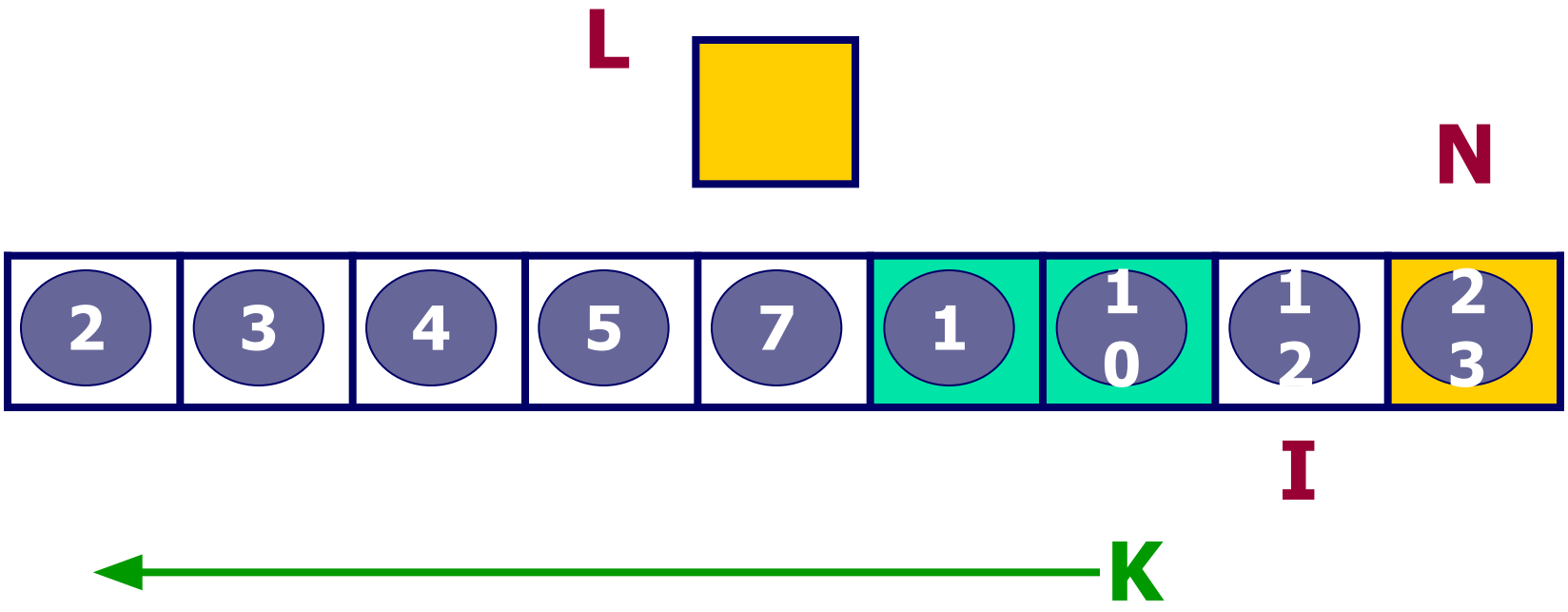
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



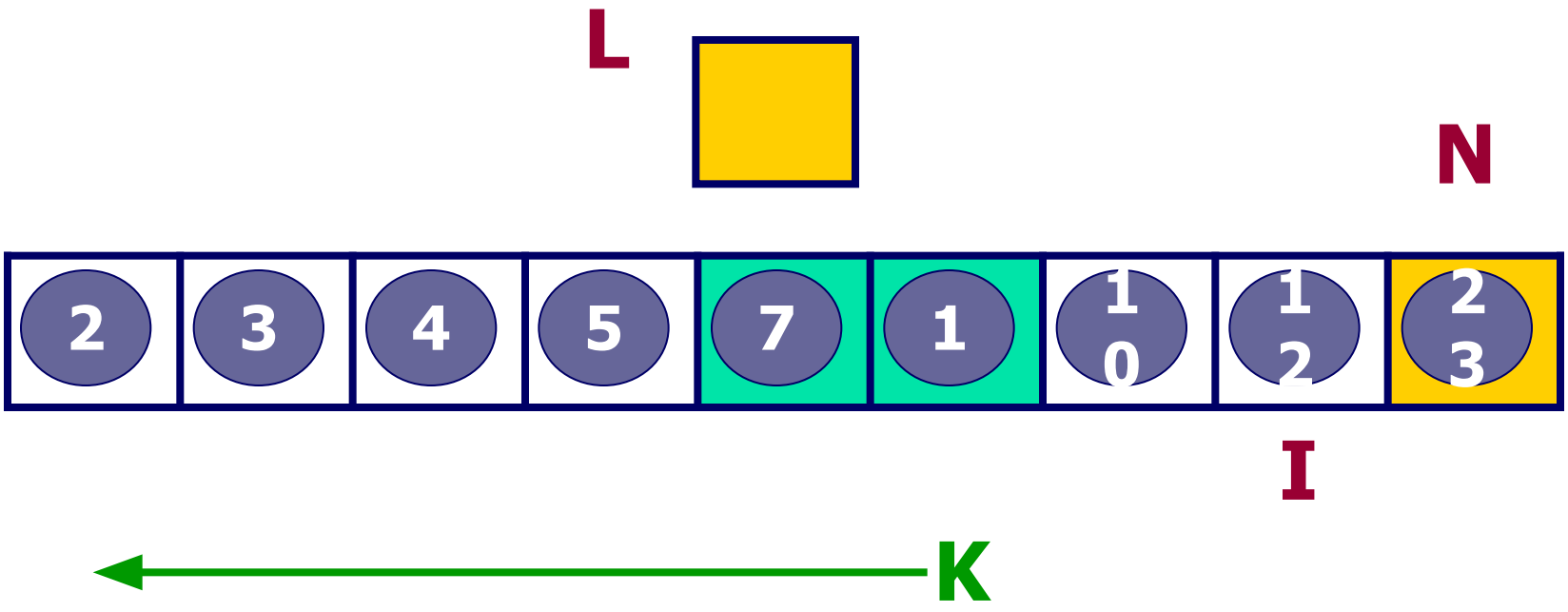
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



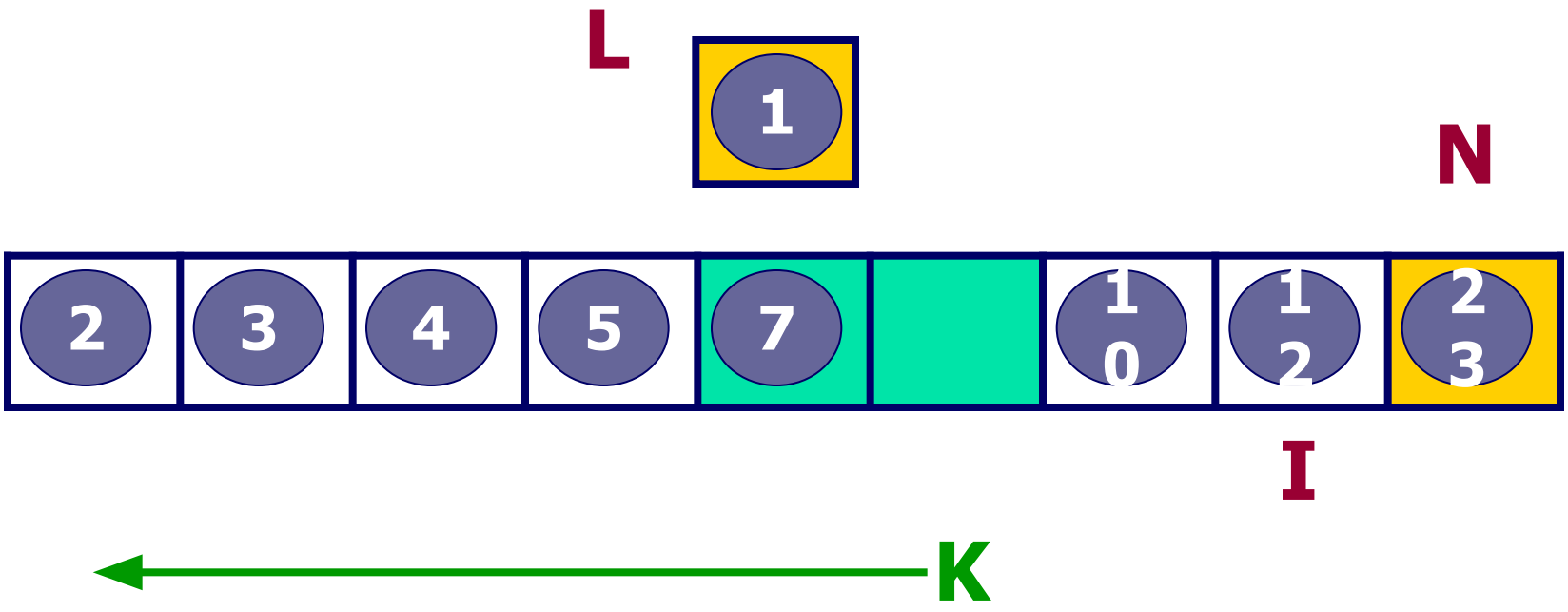
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



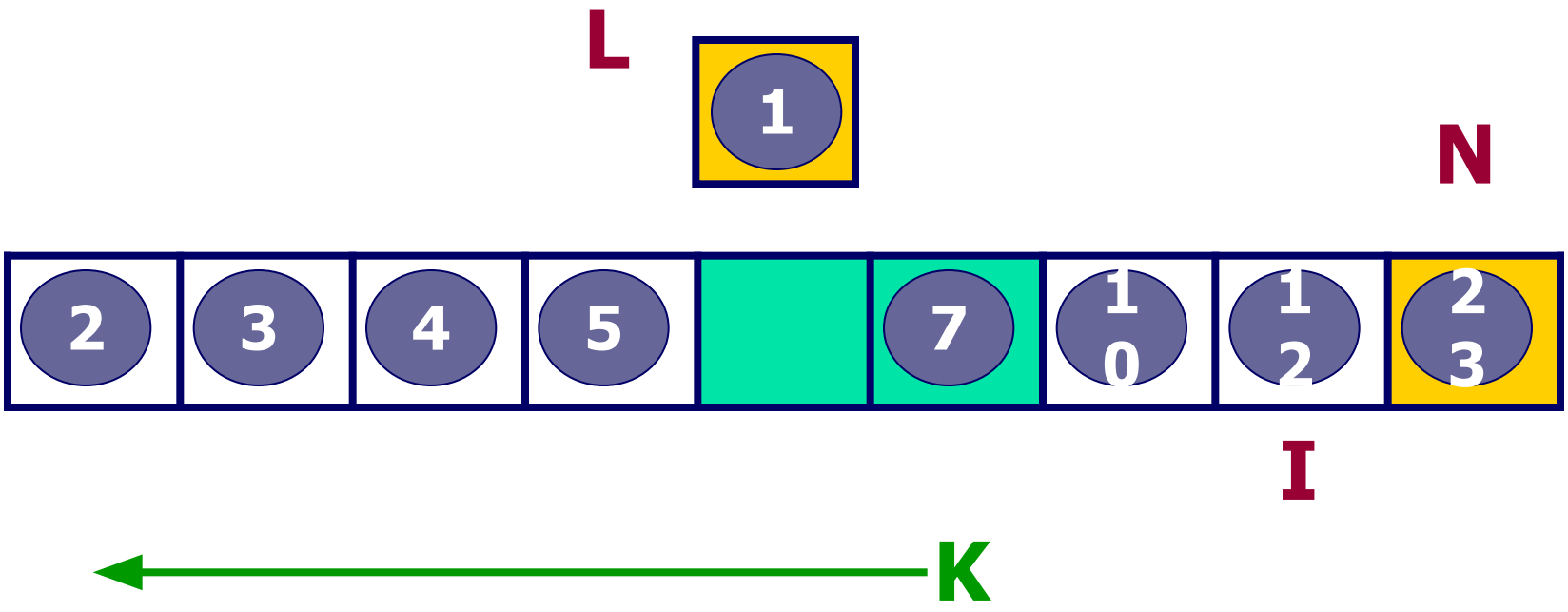
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



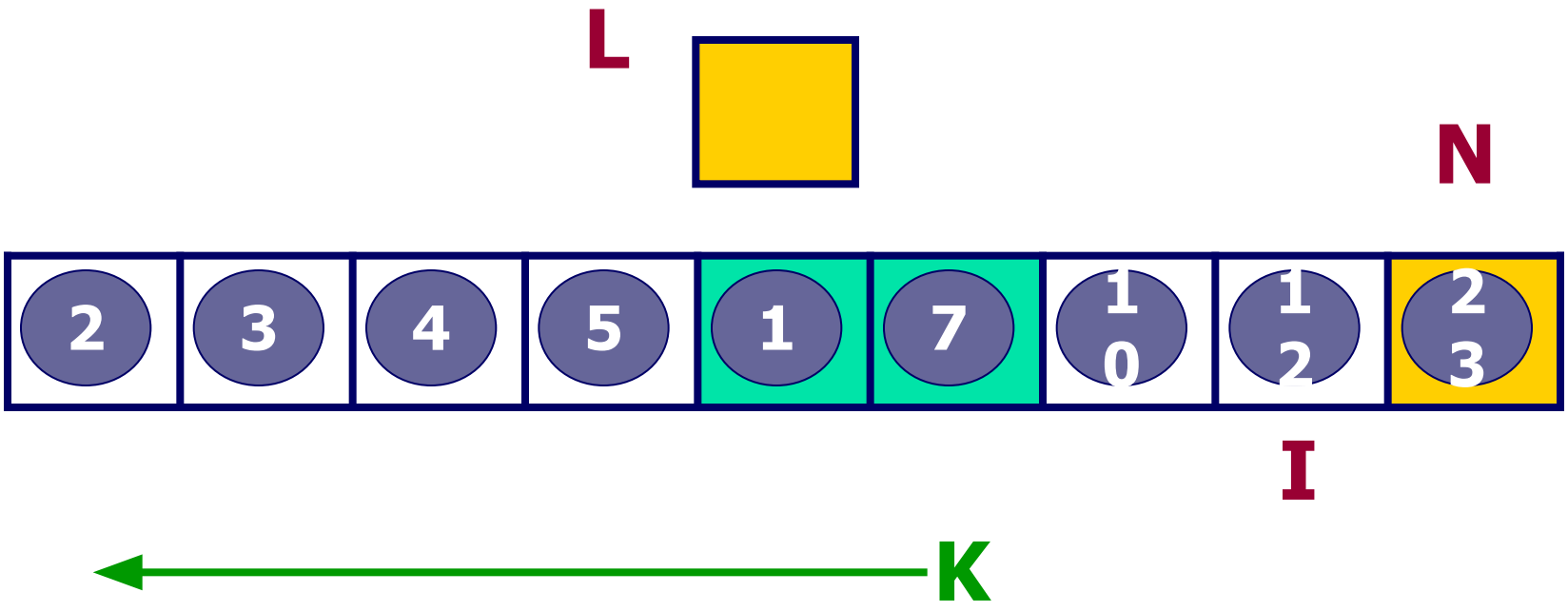
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



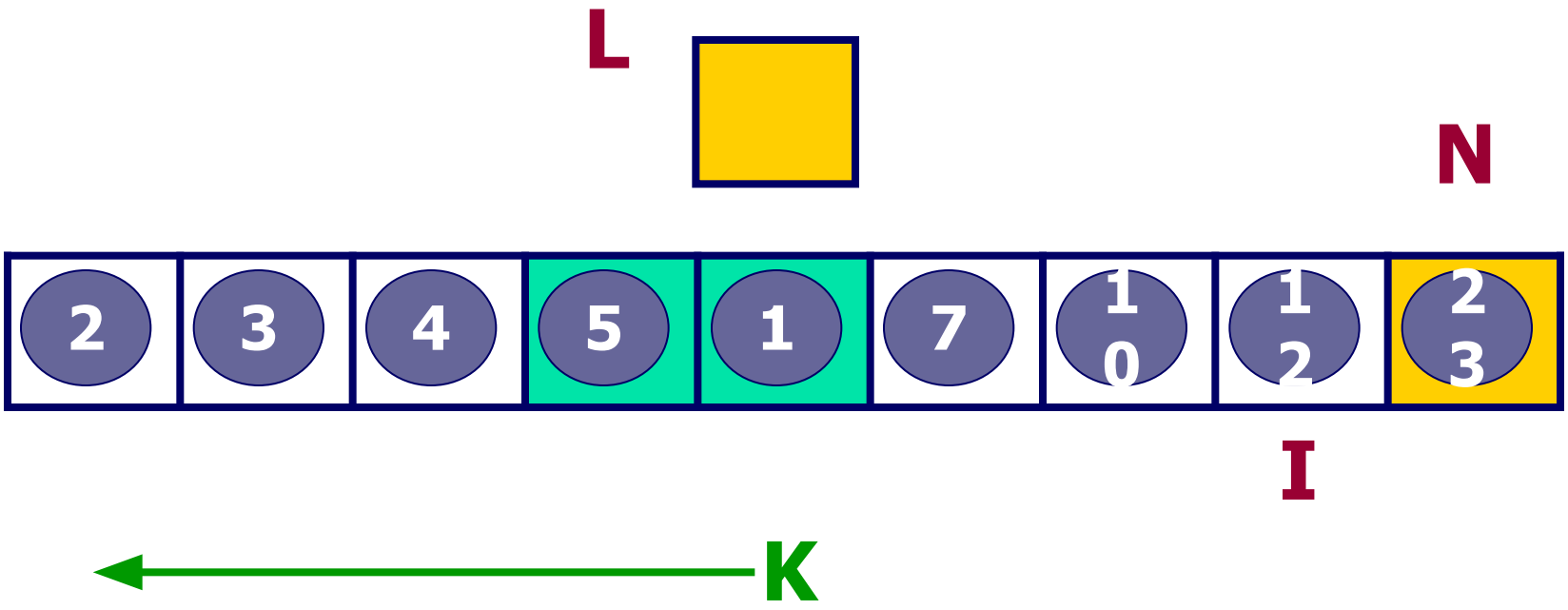
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



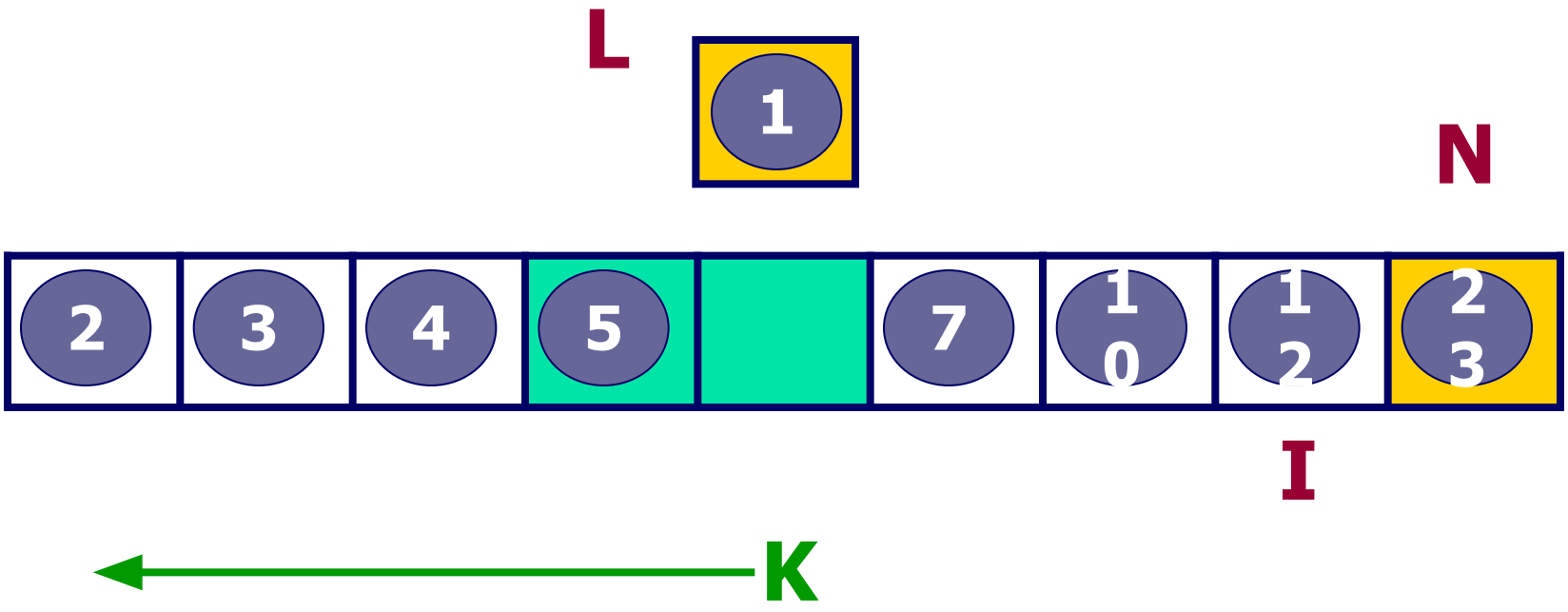
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



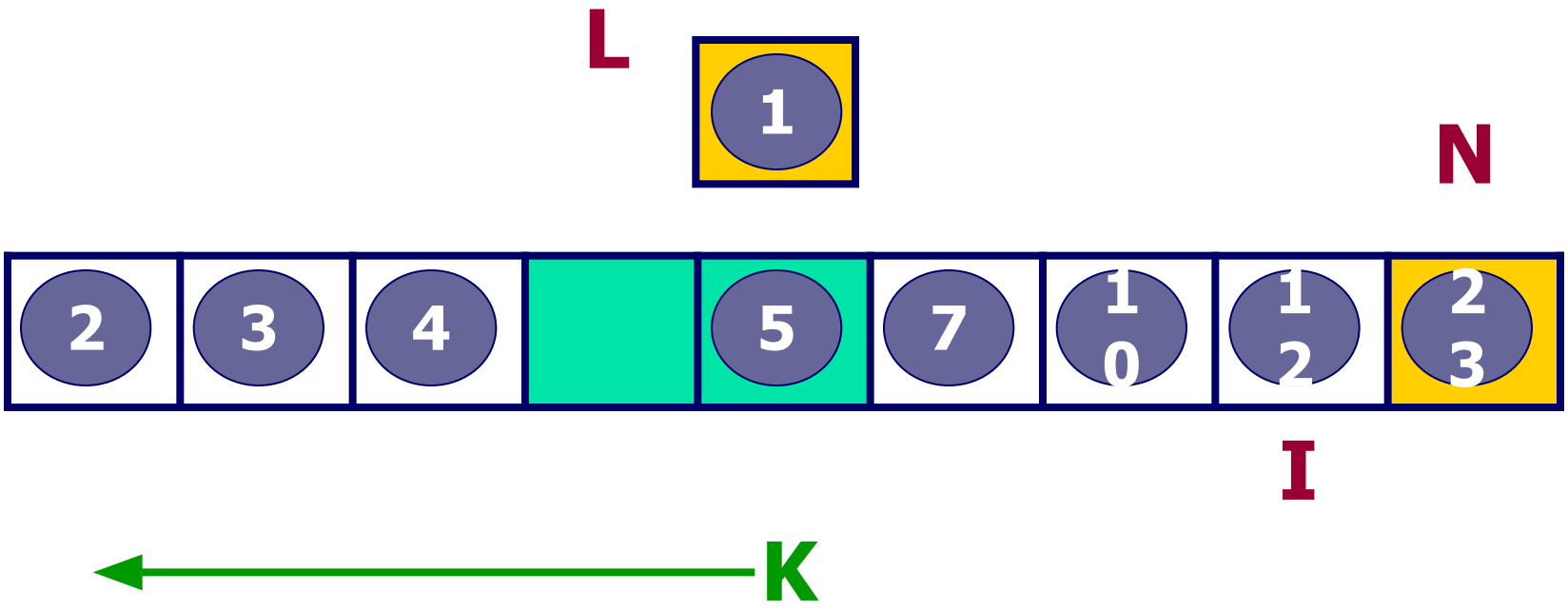
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



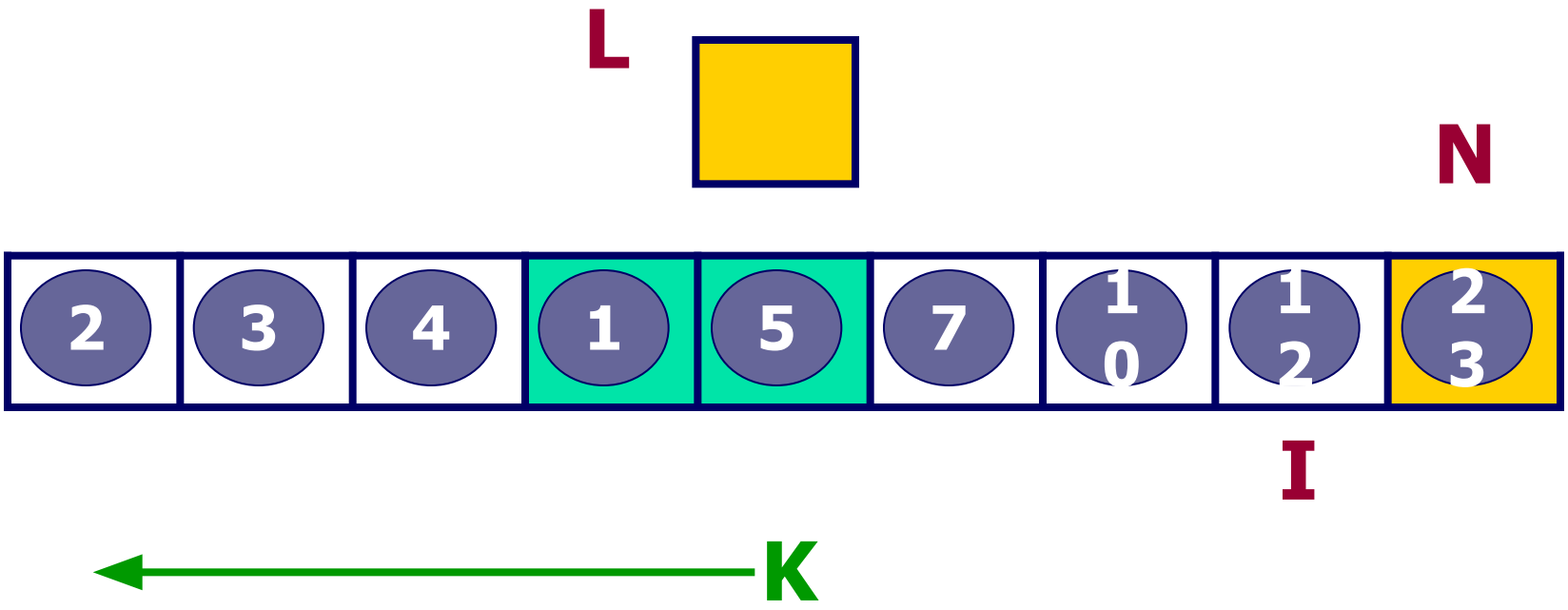
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



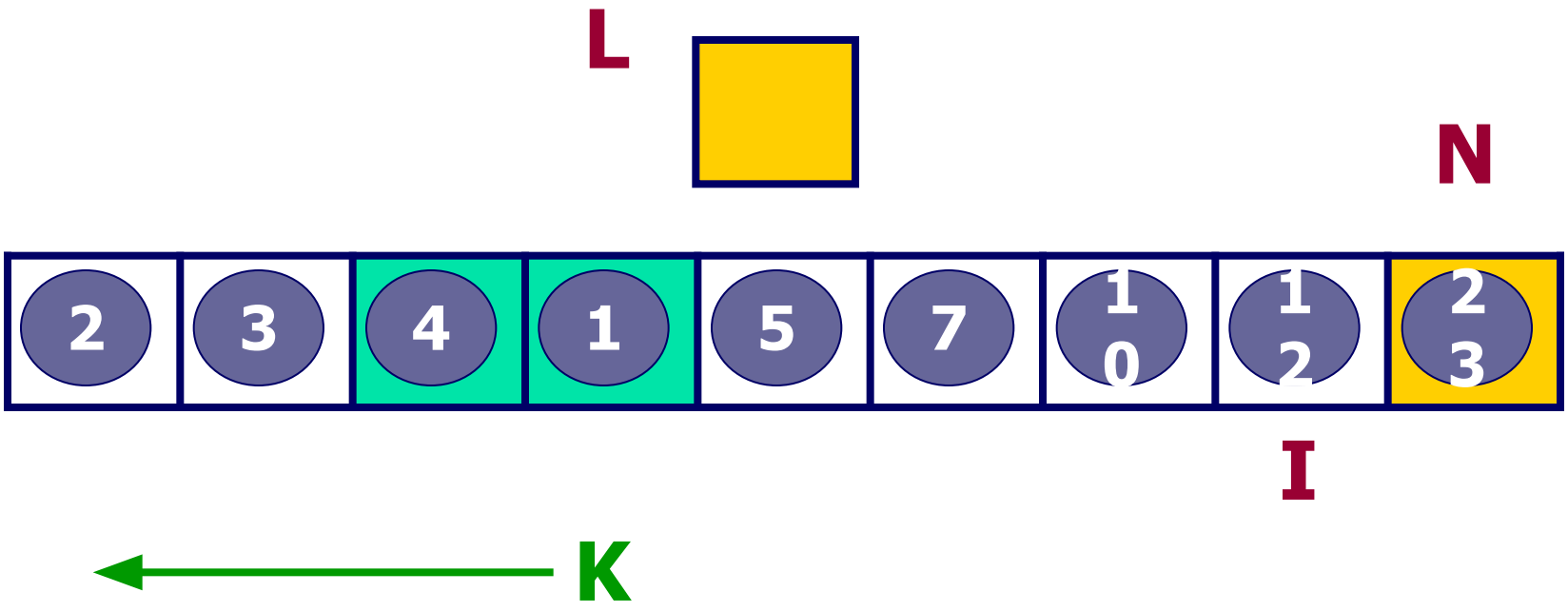
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



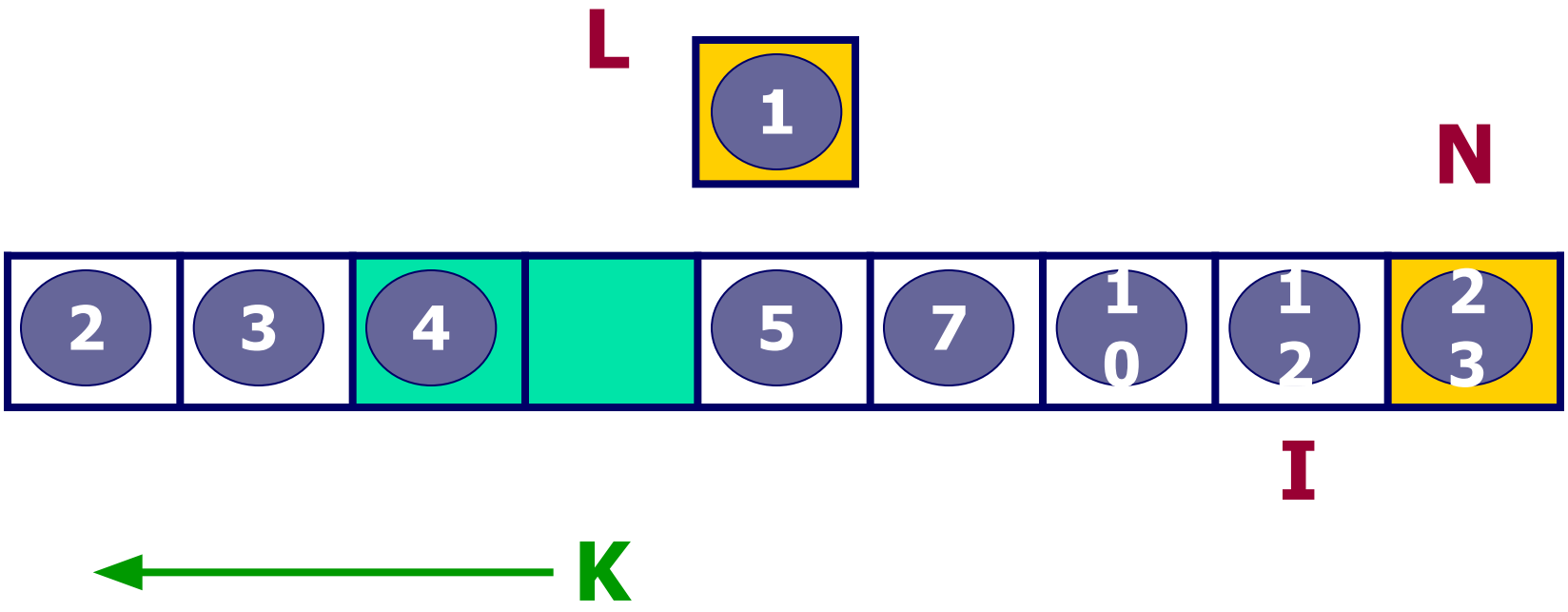
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



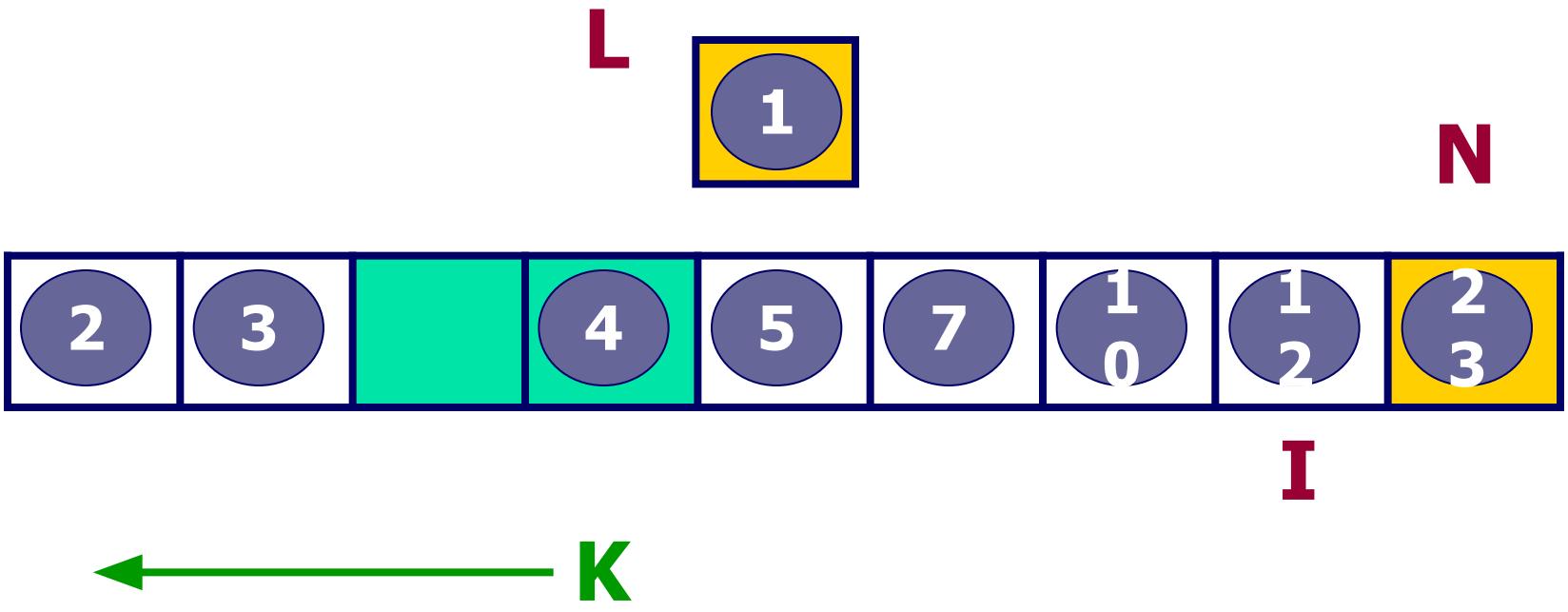
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



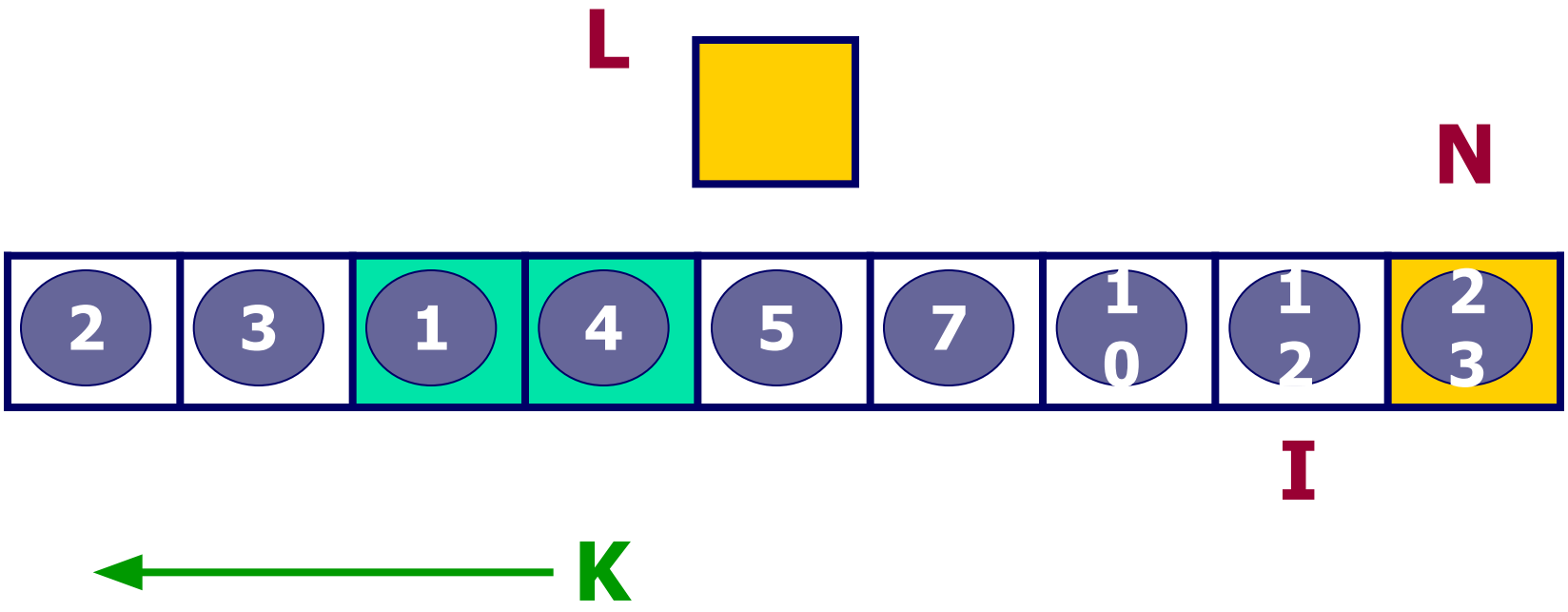
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



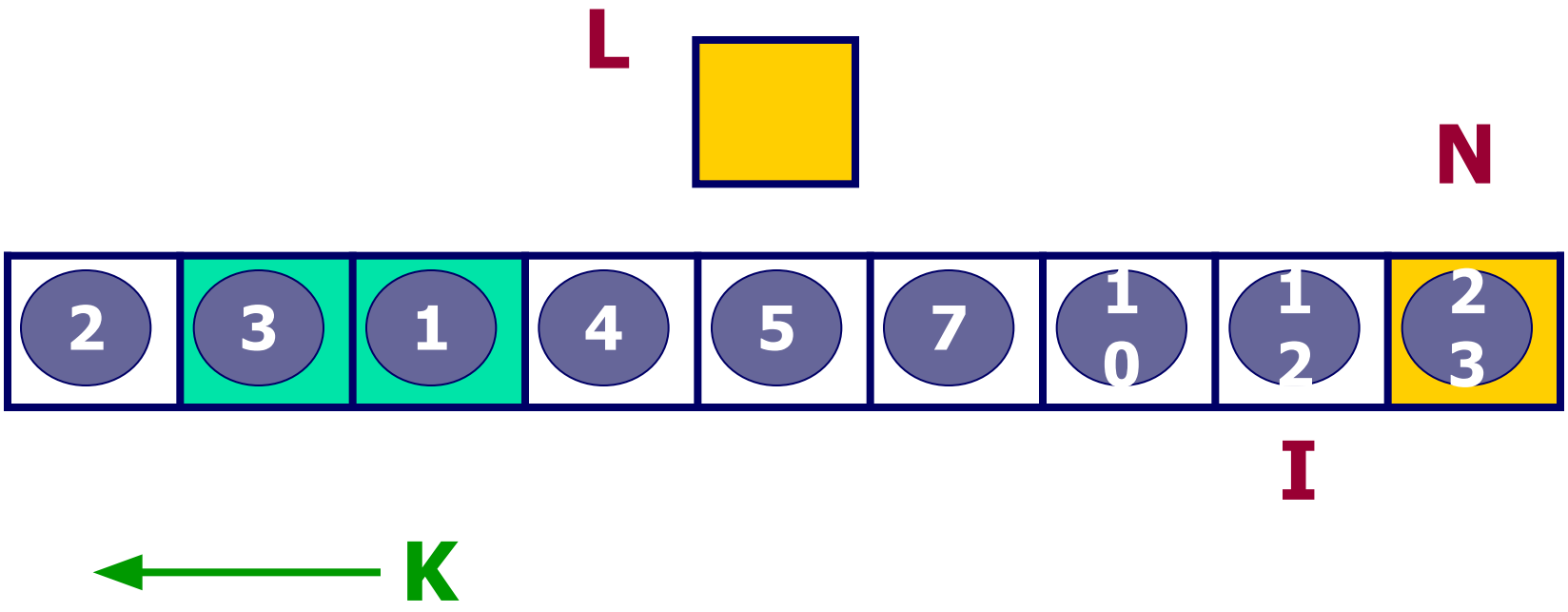
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



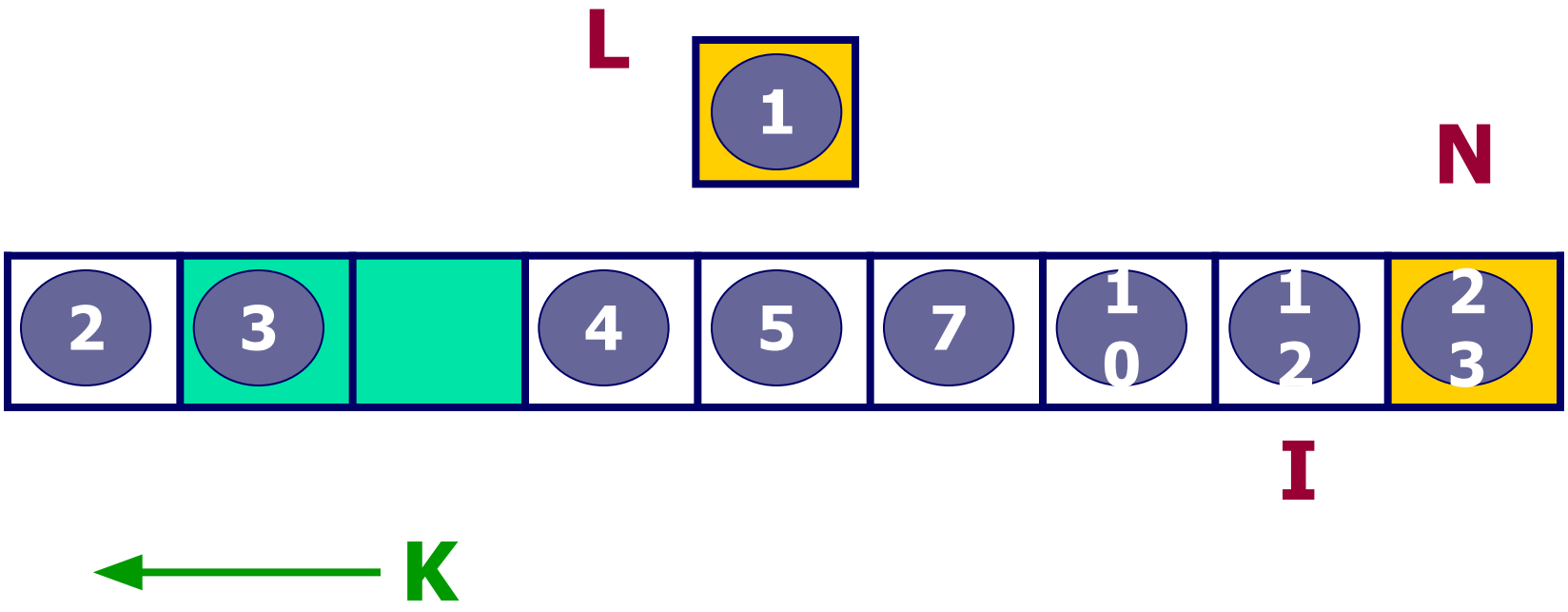
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



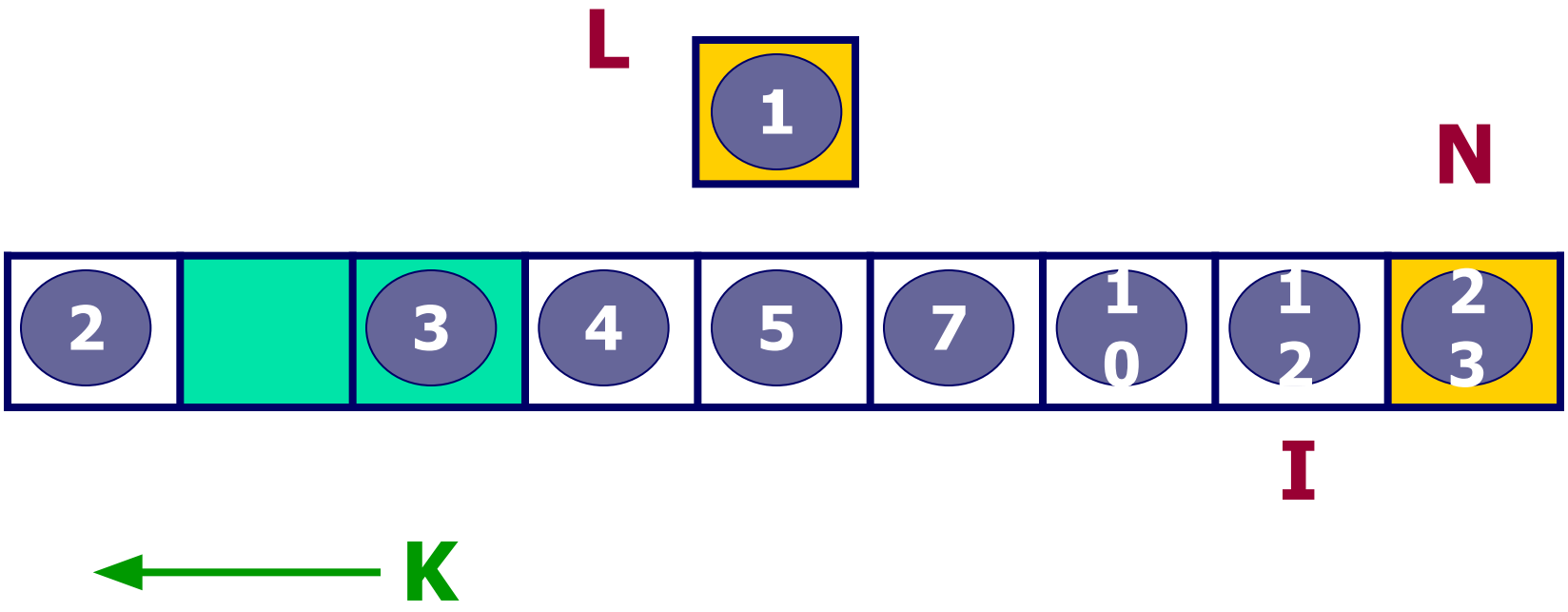
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



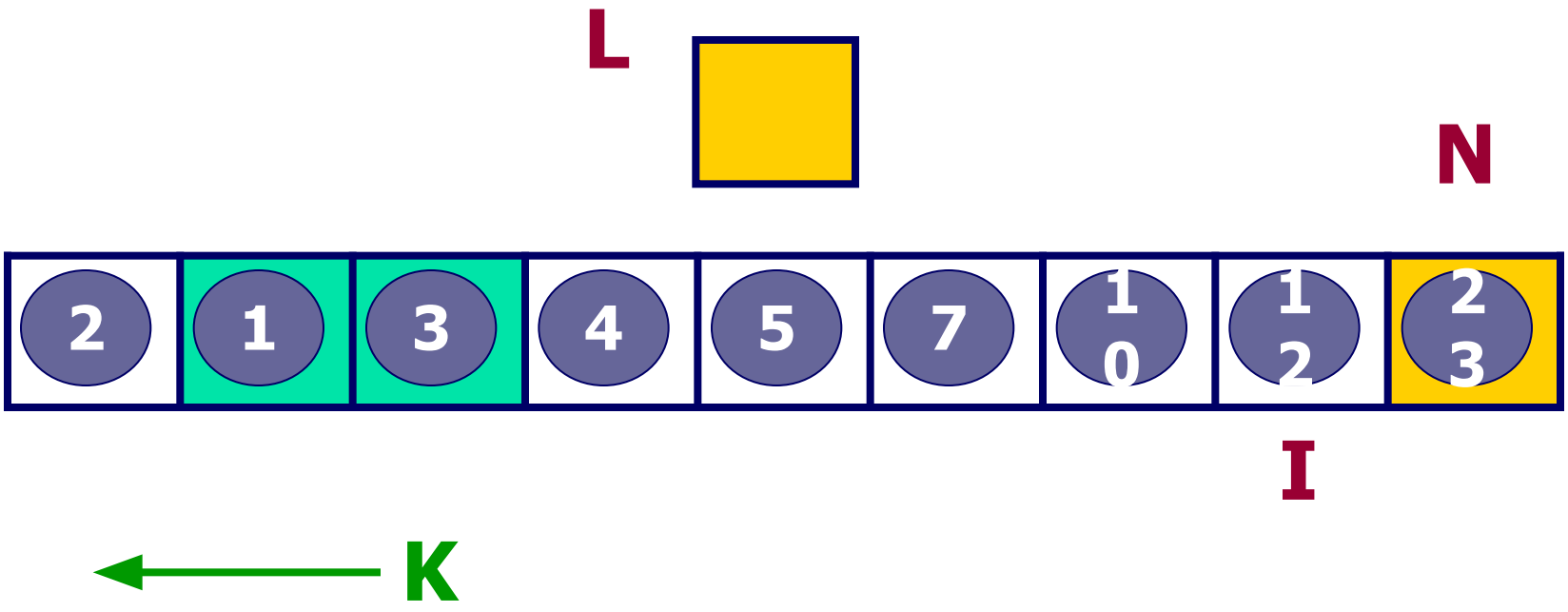
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



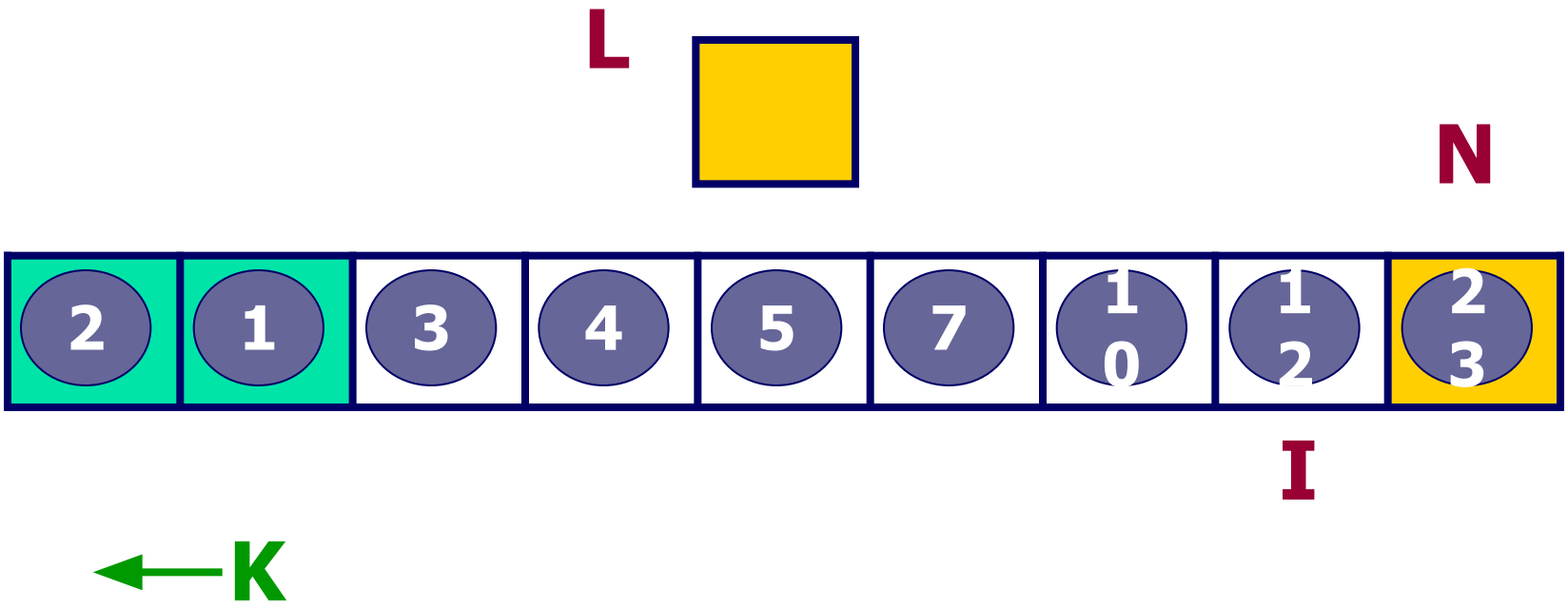
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



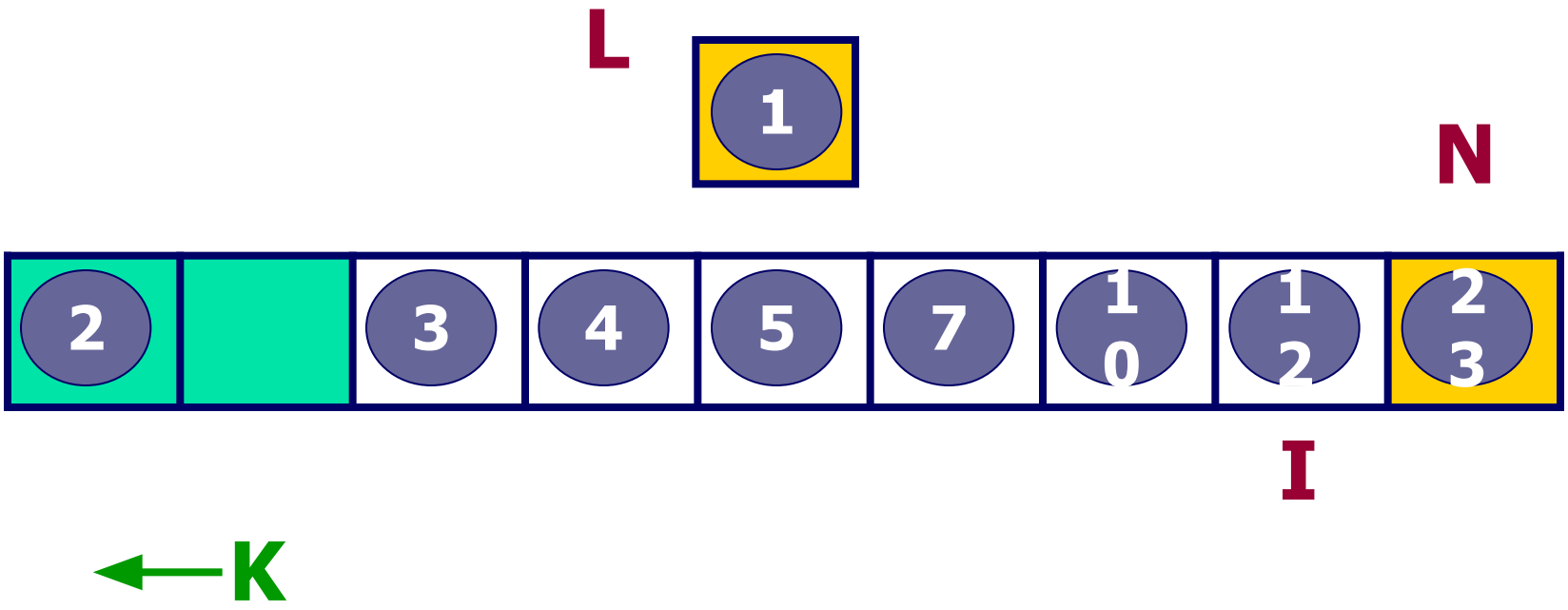
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



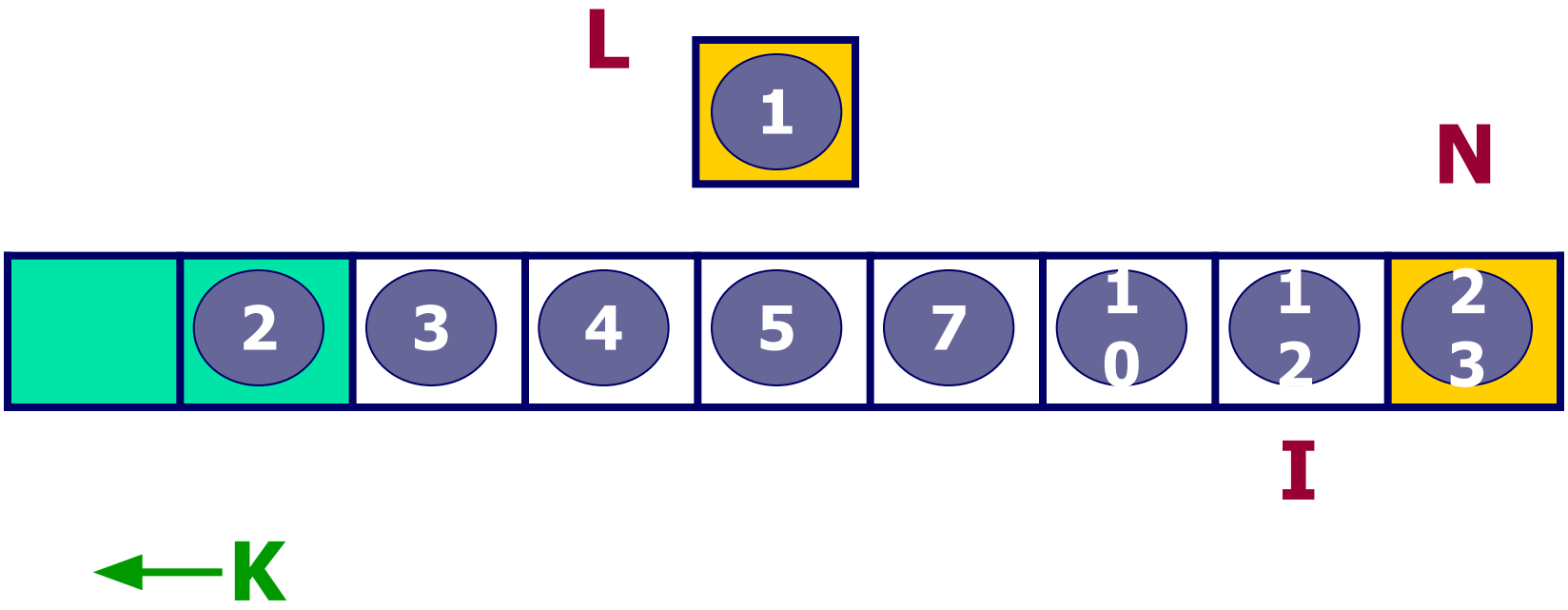
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



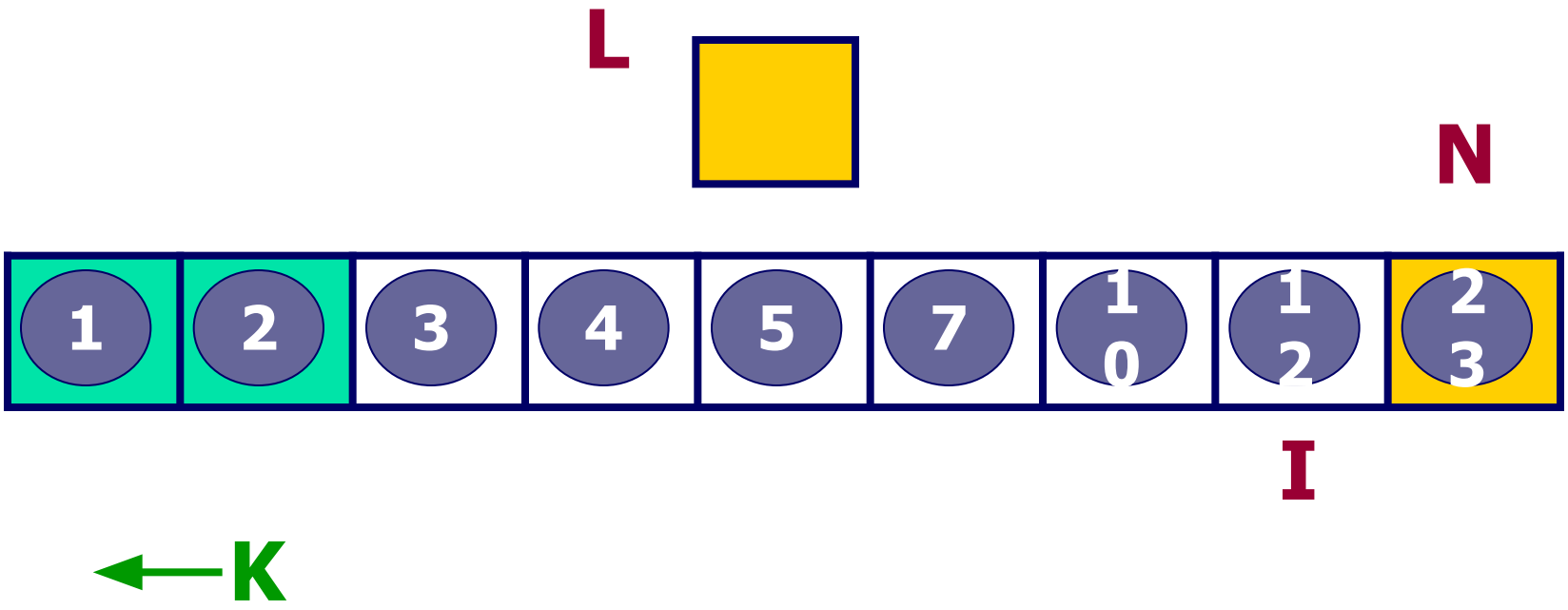
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



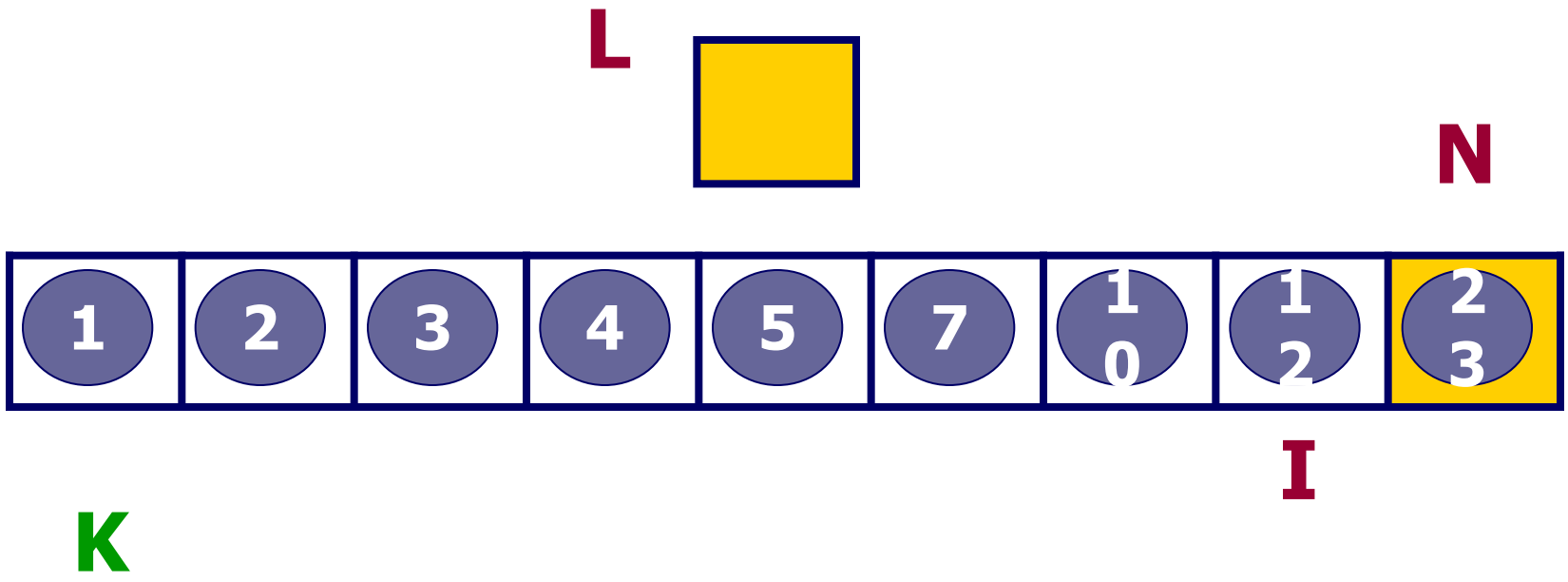
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



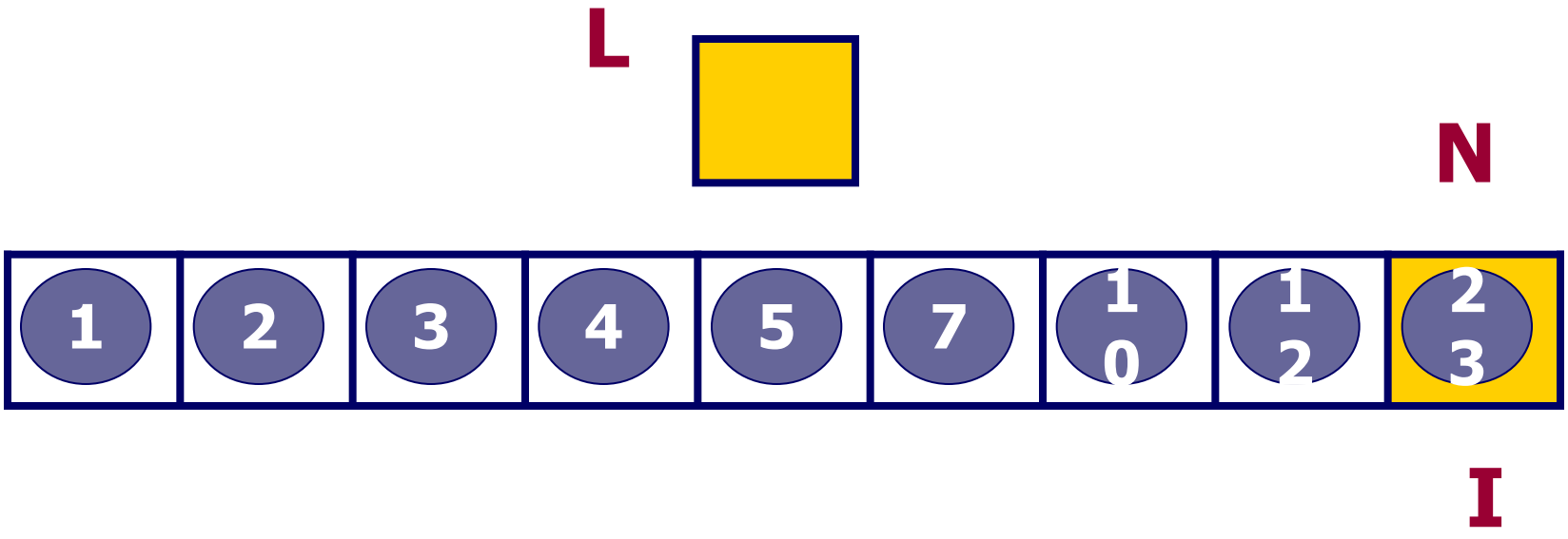
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



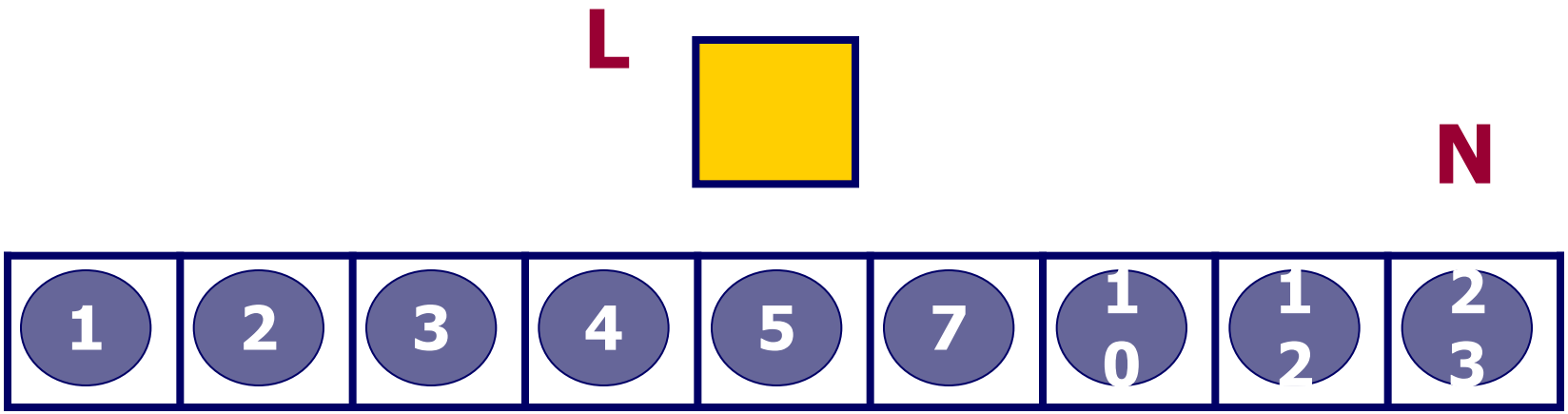
ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



ПРОДВИЖЕНИЕ ЭЛЕМЕНТОВ



Словесное описание метода

Будем сортировать массив по неубыванию значений элементов

П1 Сравнить **A[1]** и **A[2]**, если элементы стоят не в порядке сортировка, то поменять их местами (Для обмена используем вспомогательную переменную L).

П2 Предположим, что часть элементов массива от **A[1]** до **A[I]** уже упорядочена. Элемент **A[I]** сравнивается с элементом **A[I+1]**.
Если **A[I] > A[I+1],** то меняем элементы местами и переходим к **П3**.
Иначе переходим к следующей паре элементов **I:=I+1** и повторяем **П2** пока **I<=N-1** (т.е. пока не сравним два последних элемента массива).

П3 (В **П3** мы попадаем потому, что элементы **A[I]** и **A[I+1]** поменялись местами. На месте **I** оказался элемент с другим значением. Его надо поставить на своё место в упорядоченной группе элементов от **A[1]** до **A[I-1])**.
Организуем цикл в обратном направлении по переменной **K**. **K:=I**.
Если **A[K]>=A[K-1]**, то **I:=I+1** и переходим к **П2**. Иначе меняем элементы **A[K]** и **A[K-1]** местами, и переходим к следующей паре **K:=K-1**.
П3 повторяем пока не дойдем до начала массива или не перейдём к **П2**.

```
program linsort;          {заголовок программы, не обязателен}
TYPE                      {секция описания типов}
 MASS= array [1..30] of integer; {объявляется  тип}
var                       {секция описания переменных}
 N:1..30;                 {размер массива }
 A: MASS;                 {массив из N целых чисел}
 I:1..30;                 {переменная цикла  }
 L:integer;               {переменная для обмена}
 K:0..30;                 {для обратного цикла}
 CS: integer;             {счётчик числа сравнений}
 CP: integer;             {счётчик числа перестановок}
```

Порядок работы:

Разработка, отладка и тестирование программы:

Программа должна:

- Сформировать массив (ввод данных с клавиатуры);
- Вывести массив на экран для просмотра данных;
- Произвести сортировку массива по алгоритму метода «Пузырька»;
- Вывести массив на экран для просмотра результата.

После того, как Вы убедились, что программа работает правильно

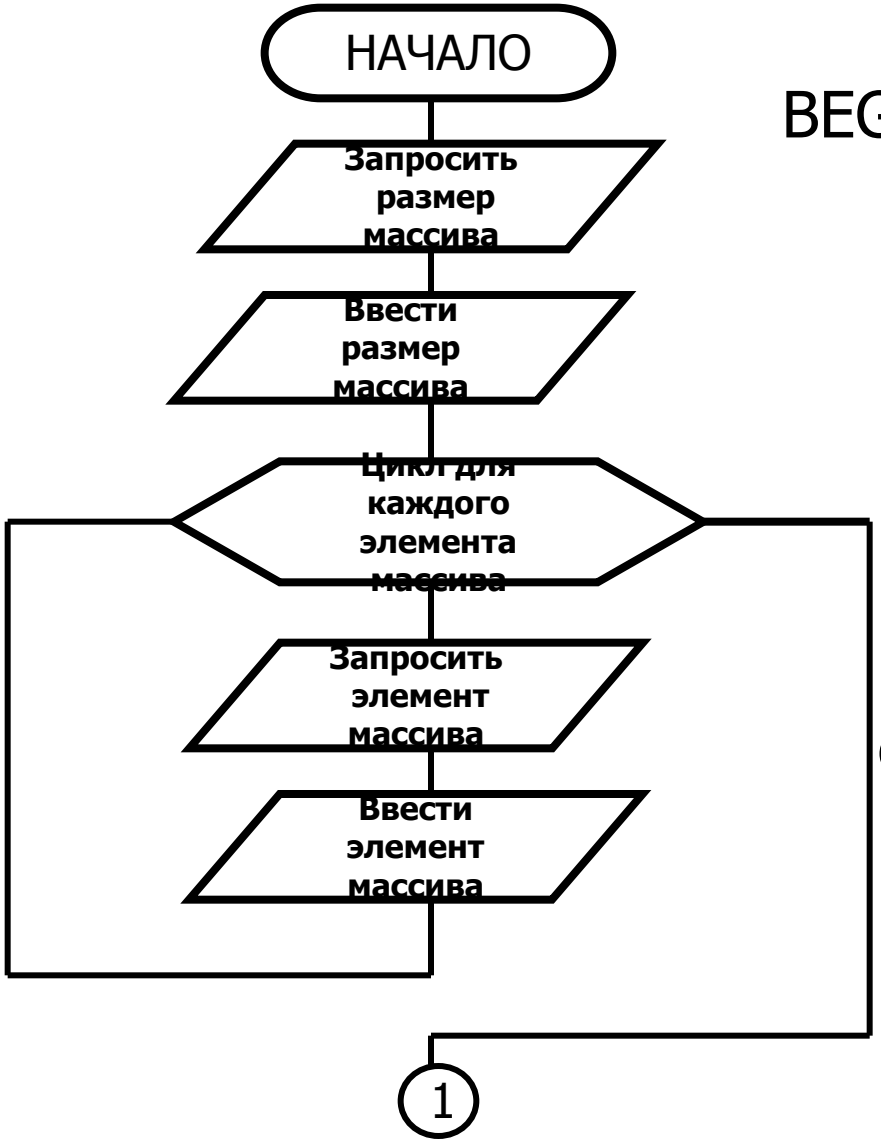
Определить эффективность метода:

- Поставить счётчики в программу;
- Запустить программу на выполнение;
- Снять показания счётчиков на первом входном массиве;
- Записать показания счётчиков в бланк лабораторной работы;
- Запустить программу и снять показания счётчиков на втором и третьем входных массивах.
- Описать дополнительное рабочее поле ОЗУ в бланке лабораторной работы.

РАЗРАБОТКА АЛГОРИТМА

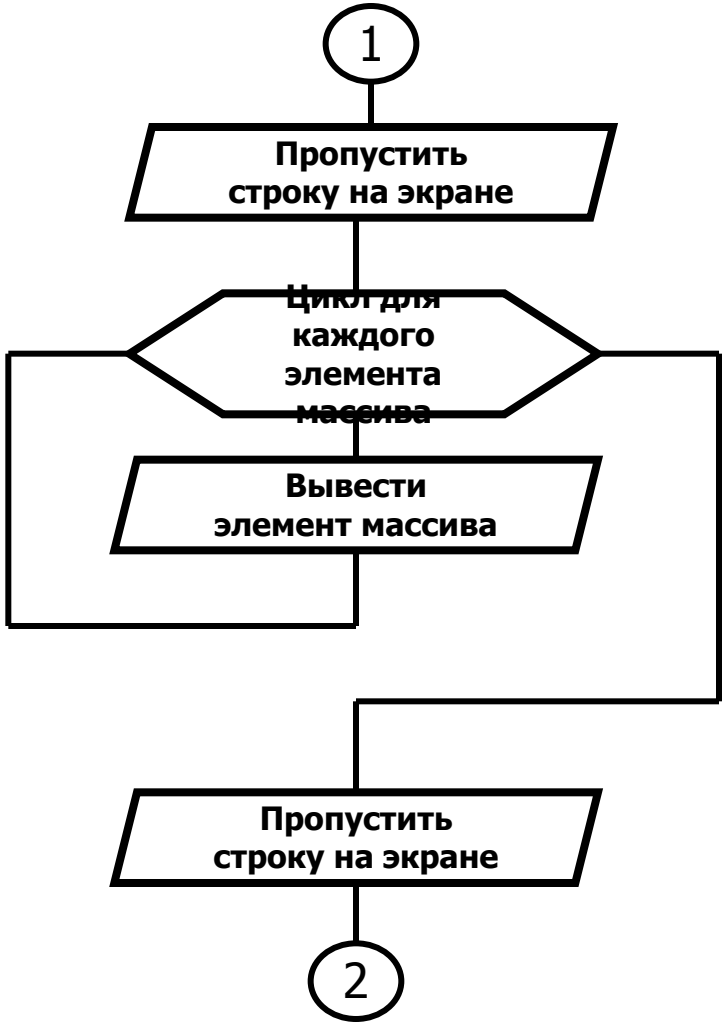
МЕТОДА «ПУЗЫРЬКА»
(массив целых чисел сортируется по неубыванию элементов)

Блок формирования массива



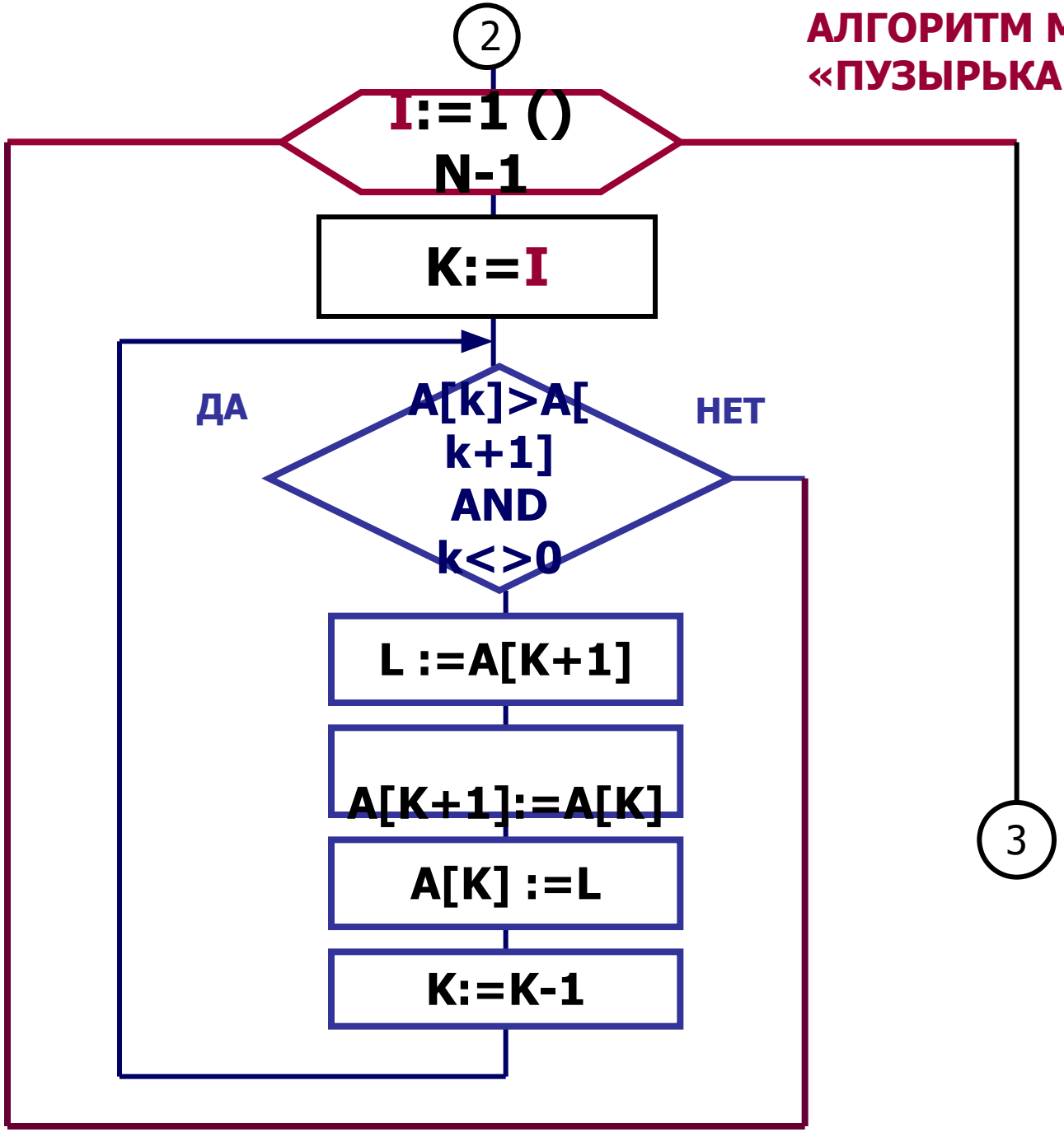

```
BEGIN
    Write(' N= ');
    ReadLn(N);
    FOR I:=1 TO N DO
begin
    Write(' A[ ' , I , ' ]= ');
     ReadLn(A[ I ])
end;
```

Блок печати массива




```
WriteLn;

FOR I:=1 TO N DO
    Write(A[ I ], ' ');

WriteLn;
```

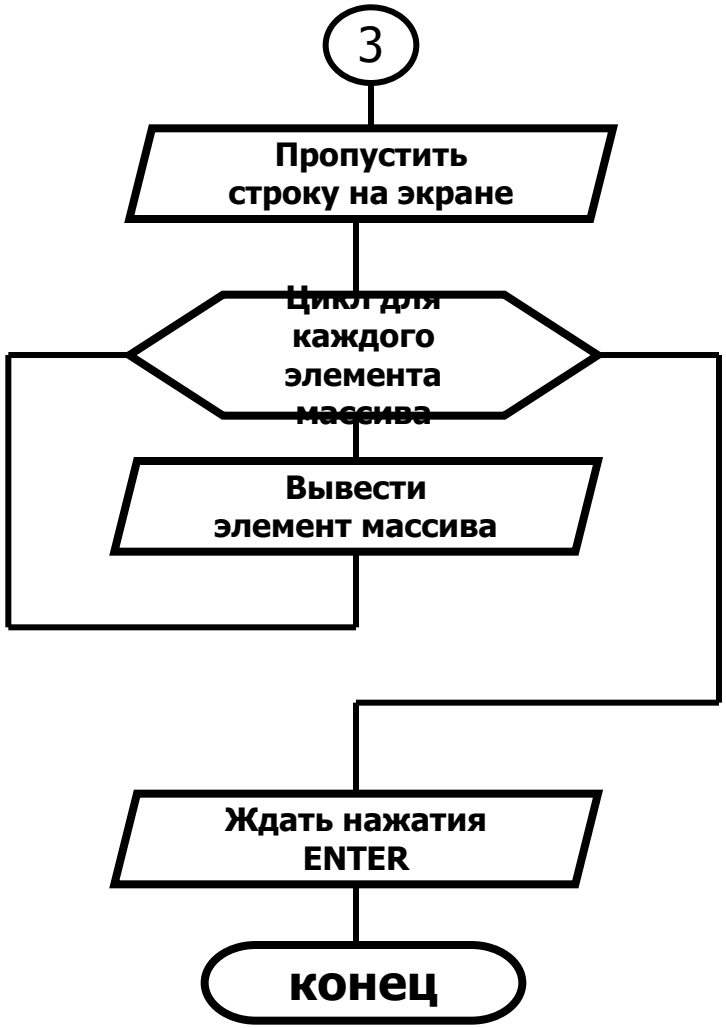
АЛГОРИТМ МЕТОДА «ПУЗЫРЬКА»



ОСНОВНАЯ ЧАСТЬ ПРОГРАММЫ

```
FOR I:=1 TO N-1 DO
    BEGIN
        K:=I;
        WHILE (A[K]>A[K+1])  AND (K<>0)
DO
            begin
                L:=A[K+1];
                A[K+1]:=A[K];
                A[K]:=L;
            K:=K-1;
            end;
    END;
```

После завершения сортировки ещё раз вывести на экран значения элементов массива, чтобы проверить, что сортировка прошла успешно.




```
WriteLn;
FOR I:=1 TO N DO
    Write(A[ I ] , ’  ’);
 ReadLn;
```

END. { конец программы}

Куда ставить счётчики?

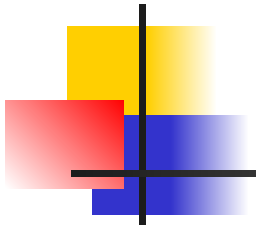
```
CS:=0;
CP:=0;
FOR I:=1 TO N-1 DO
    BEGIN
        K:=I;
        CS:=CS+1;
        WHILE (A[K]>A[K+1] ) AND (K<>0)  DO
            begin
                CS:=CS+1;
                L:=A[K+1];
                A[K+1]:=A[K];
                A[K]:=L;
            K:=K-1;
                CP:=CP+3;
            end;
    END;

WriteLn(' CS=' ,CS);
WriteLn(' CP=' ,CP);
```

Обнулить счётчики до начала сортировки

Увеличить на 1 значение счётчика числа сравнений

Увеличить на 3 значение счётчика числа перестановок

Вывести на экран значения счётчиков после завершения сортировки



Внимание!

Переменные-счётчики нужны только для проведения эксперимента.

Они не влияют на алгоритм сортировки и во время сортировки не задействованы.

Эти переменные не должны учитываться как дополнительная рабочая память.