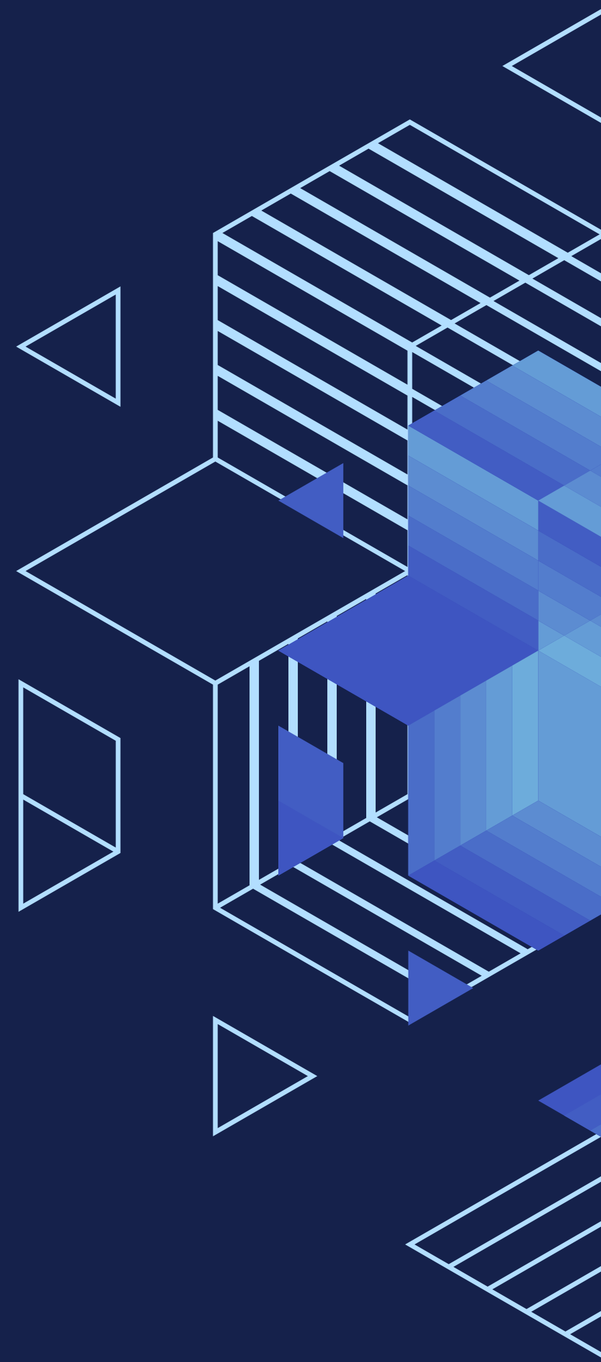

SHELL SORT
SORTING ALGORITHM

Table Of Content

- Brief History
  Who and when was Shell Sort Algorithm developed
- Original Version
  The first ever version of Shell Sort Algorithm
- Variations
  Different ways to implement Shell Sort Algorithm
- Complexity
  Time and Space Complexity of Shell Sort Algorithm
- Simulation
  Let's visualize Shell Sort Algortihm
- Internet vs Streamline Code



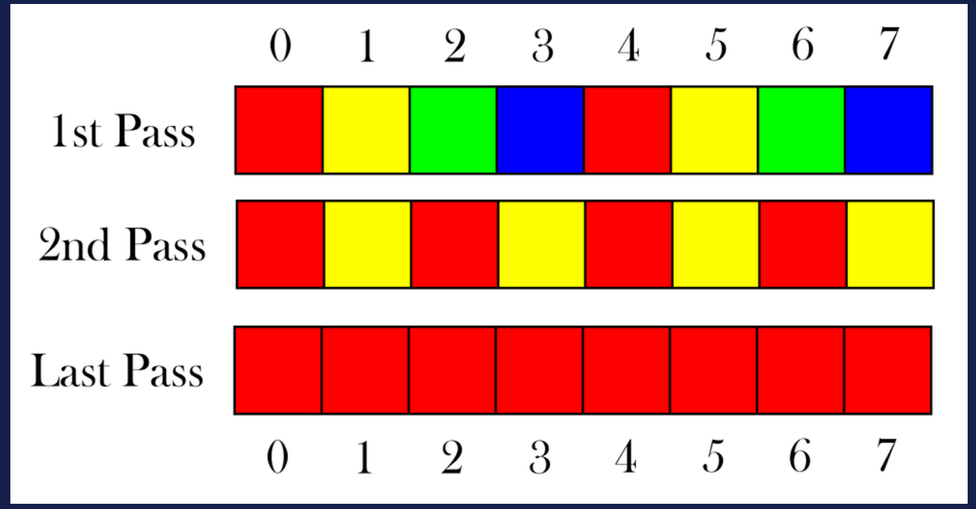
Shell Sort

- Shel Sort Algorithm was invented by Donald Shell in 1959.
- He acquired his Ph.D. in Mathematics from the University of Cincinnati in 1959, after publishing the shell sort algorithm in the Communications of the ACM in July the same year.

Gap Sequence

- In Shell Sort, the sorting process is done in intervals.
- The interval between the elements in each pass is reduced based on the **gap sequence** used.
- Choosing the optimal gap sequence can help in improving the efficiency of the process.

Shell's original sequence	N/2 , N/4 , …, 1
Knuth's increments:	1, 4, 13, …, (3k – 1) / 2



Time and Space Complexity of Shell Sort

Best Case Time Complexity	O(n log n)
Average Time Complexity	depends on gap sequence
Worst Case Time Complexity	O(n^(4/3))
Space Complexity	O(1) (Sort in Place)

Source: https://cheatography.com/pryl/cheat-sheets/sorting-algorithms/

Shell Sort

- Shell sort is a generalized version of the **insertion sort algorithm**. It first sorts elements that are far apart from each other and successively reduces the interval between the elements to be sorted. *(Programwiz.com)*

- It is essentially several stages of Insertion Sort designed with the purpose of speeding up to the said algorithm itself. (Mahmoud, 2000)

Shell Sort

- PROS
  - Shell Sort runs faster than an insertion sort
  - It improves upon bubble sort and insertion sort by moving out of order elements more than one position at a time.
  - Shell Sort is efficient for medium size list of elements.

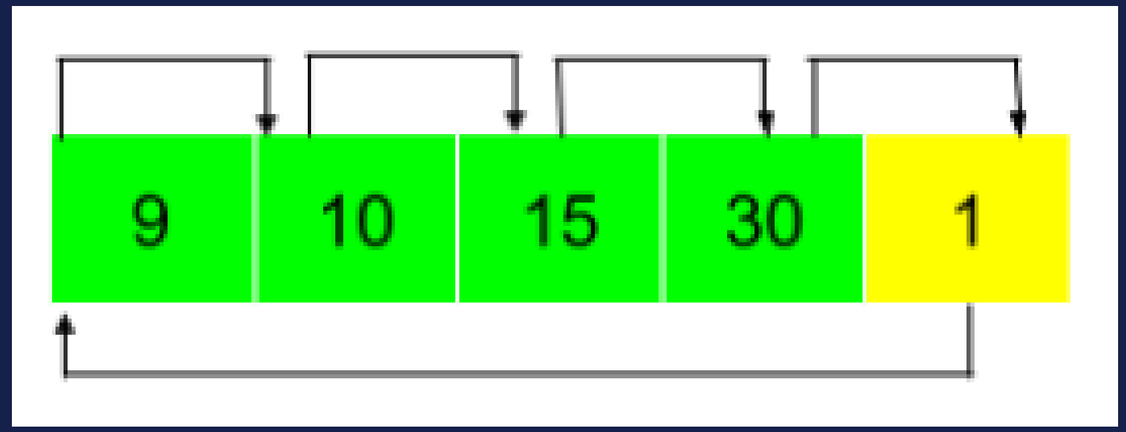
- Cons
  - Shell sort algorithm is not stable method, as its performance depends on the choice of increment sequence.

Source: Chhatwani, P. K. (2014). Insertion Sort with its Enhancement . International Journal of Computer Science and Mobile Computing , 3(3), 801–806.

Variation 0 (Insertion Sort)

```
void insertionSort(int array[], int n) {
   int interval, i, j, temp;
   for(i = 1; i < n; i++){
     temp = array[i];
     for(j = i; j > 0 && array[j-1] > temp; j--){
        array[j] = array[j-1];
     }
     array[j] = temp;
   }
}
```

Source: Shell Sort Algorithm - Programiz.com

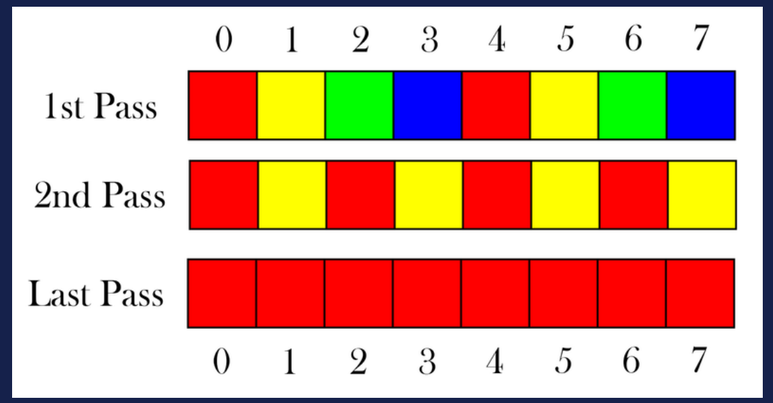


- Uses a gap of 1. It compares adjacent elements.
- The worst case for Insertion Sort is when the array is sorted in inverse order.

Variation 1 (Shell's Sequence)

```
void shellSortv1(int array[], int n) {
   int interval, i, j, temp;
   for (interval = n / 2; interval > 0; interval /= 2) {
     for (i = interval; i < n; i++) {
        temp = array[i];
        for (j = i; j >= interval && array[j - interval] > temp; j -= interval) {
          array[j] = array[j - interval];
        }
        array[j] = temp;
     }
   }
}
```

Source: Shell Sort Algorithm - Programiz.com



- Uses a gap sequence of (N / 2^p) , where p is the number of passes of sorting.
- Worst case scenario occurs for N equal to a power of two when elements greater and smaller than the median occupy odd and even positions respectively.

Variation 2 (Knuth's Interval)

```
void shellSortv2(int array[], int n) {
   int interval, i, j, temp;
   for(interval = 1; interval < n; interval = 3 * interval + 1);
   // 1, 4, 13, 40, 121, 364, 1093, ...
   for (; interval > 0; interval = interval / 3) {
     for (i = interval; i < n; i++) {
        temp = array[i];
        for (j = i; j >= interval && array[j - interval] > temp; j -= interval) {
          array[j] = array[j - interval];
        }
        array[j] = temp;
     }
   }
}
```

Source: Algorithms 4th Edition - Sedgewick & Wayne

- Uses a gap sequence of 1, 4, 13, 40, 121, ..., 3h(sub(p-1)) + 1 , where h(sub(p-1)) is the previous element of the sequence.
- Recommended for N < 1000. (Knuth, 1998)

Shell Sort interval = 5 swaps = 0

```
void shellSort(int array[], int n) {
  // Rearrange elements at each n/2, n/4, n/8, ... intervals
  for (int interval = n / 2; interval > 0; interval /= 2) {
    for (int i = interval; i < n; i += 1) {
      int temp = array[i];
      int j;
      for (j = i; j >= interval && array[j - interval] > temp; j -= interval) {
        array[j] = array[j - interval];
      }
      array[j] = temp;
    }
  }
}
```

Internet Code

```
void shellSortGroup(int array[], int n) {
   int interval, i, j, temp;
   for(interval = 1; interval < n; interval = 3 * interval + 1);
   for (; interval > 0; interval = interval / 3) {
     for (i = interval; i < n; i++) {
        temp = array[i];
        for (j = i; j >= interval && array[j - interval] > temp; j -= interval) {
          array[j] = array[j - interval];
        }
        array[j] = temp;
     }
   }
}
```

Streamlined Code

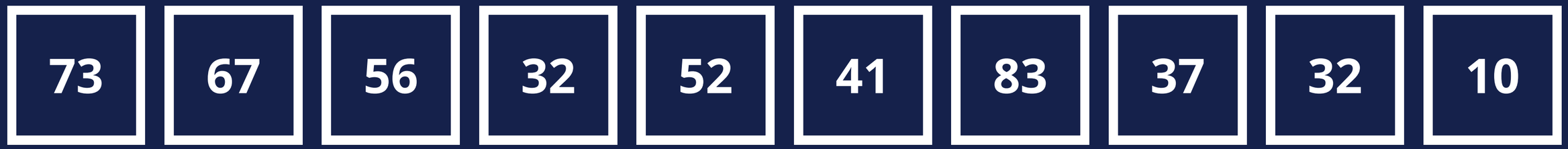
References

- https://www.programiz.com/dsa/shell-sort
- Mahmoud, H. M. (2000). Sorting: A distribution theory. John Wiley & Sons.
- https://history.computer.org/pioneers/shell.html
- https://www.cs.wcupa.edu/rkline/ds/shell-comparison.html#shellsort
- https://cheatography.com/pryl/cheat-sheets/sorting-algorithms/
- Sedgewick, R. & Wayne K. (2011). Algorithms 4th Edition
- Chhatwani, P. K. (2014). Insertion Sort with its Enhancement . International Journal of Computer Science and Mobile Computing , 3(3), 801–806.

Shell Sort

interval = 5

swaps = 0

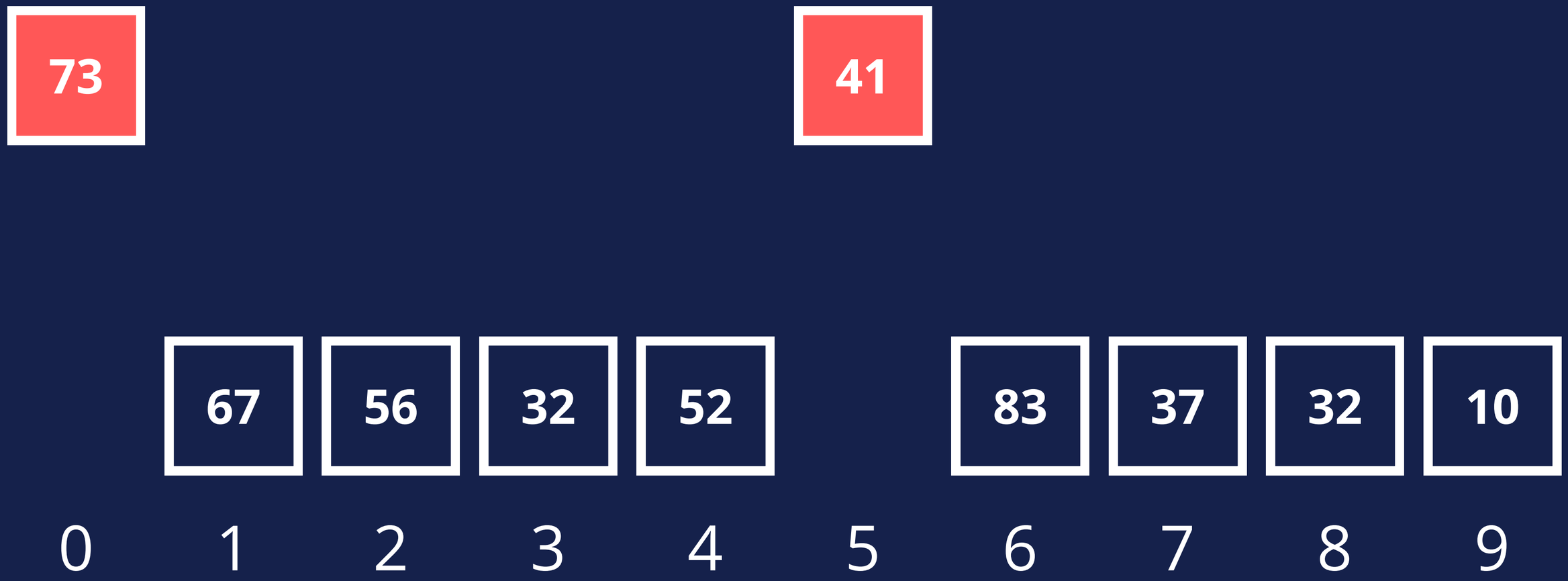


0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 5

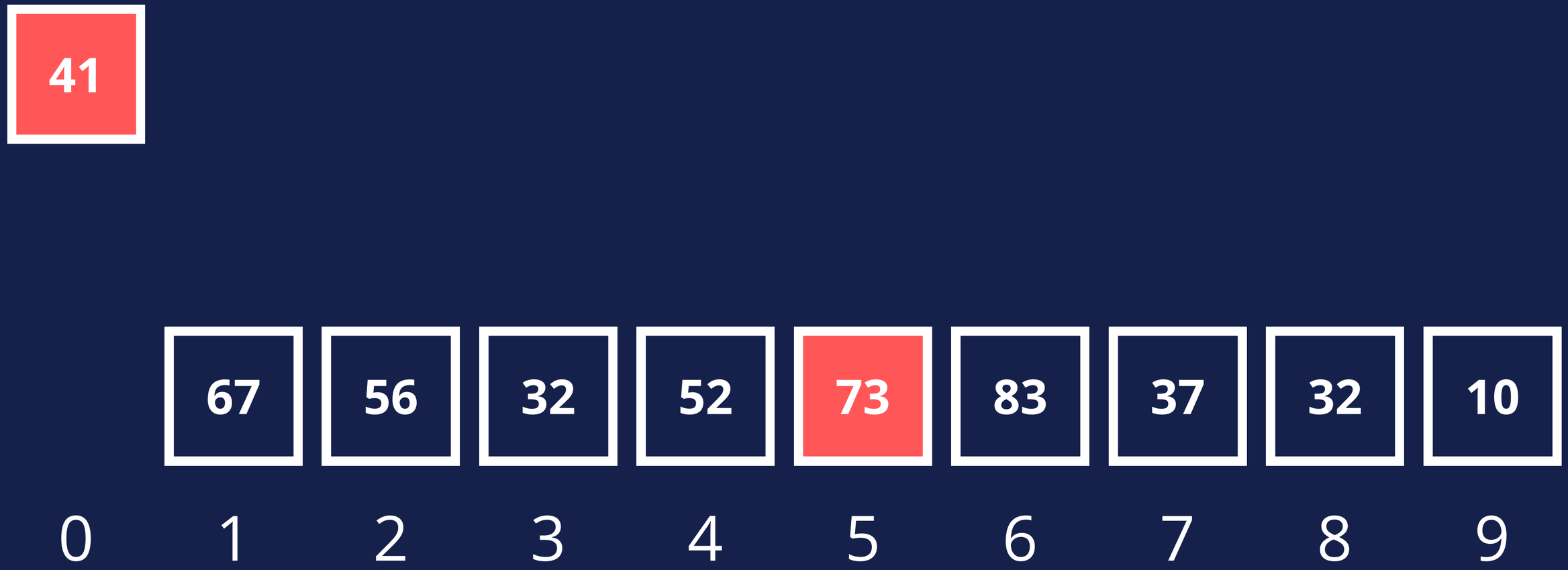
swaps = 0



Shell Sort

interval = 5

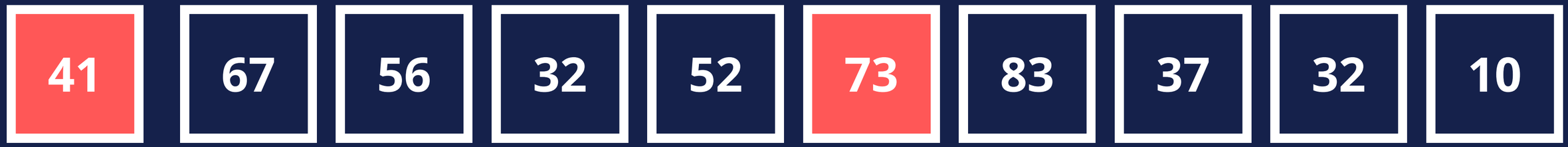
swaps = 1



Shell Sort

interval = 5

swaps = 1

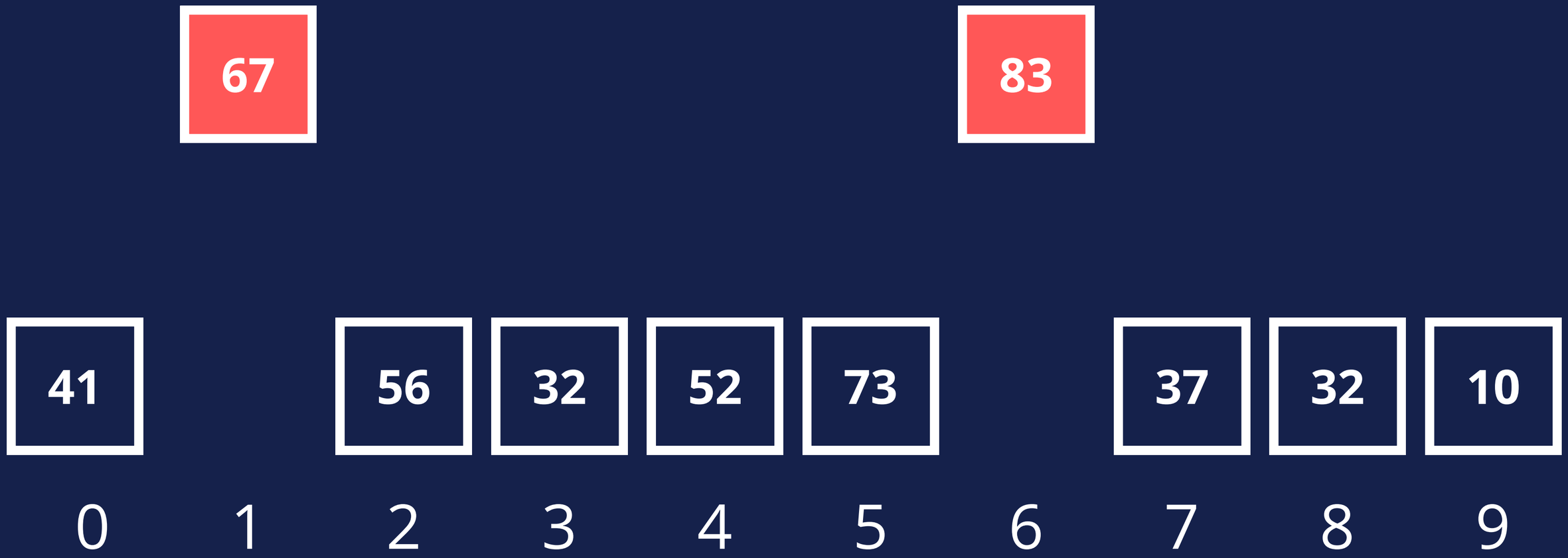


0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 5

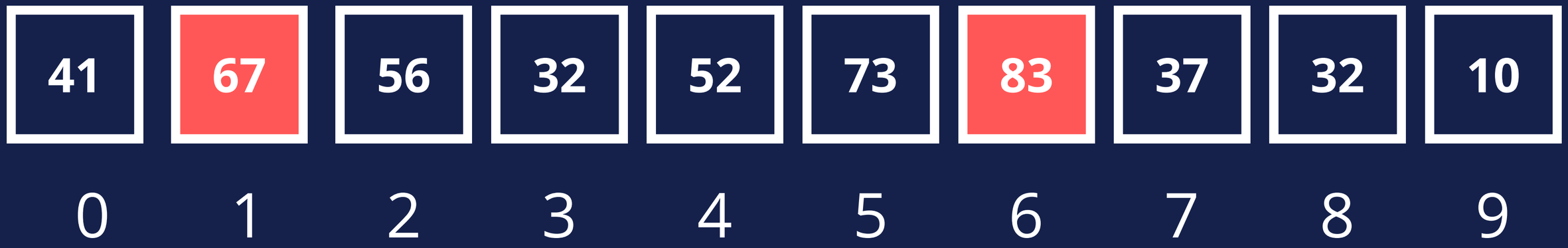
swaps = 1



Shell Sort

interval = 5

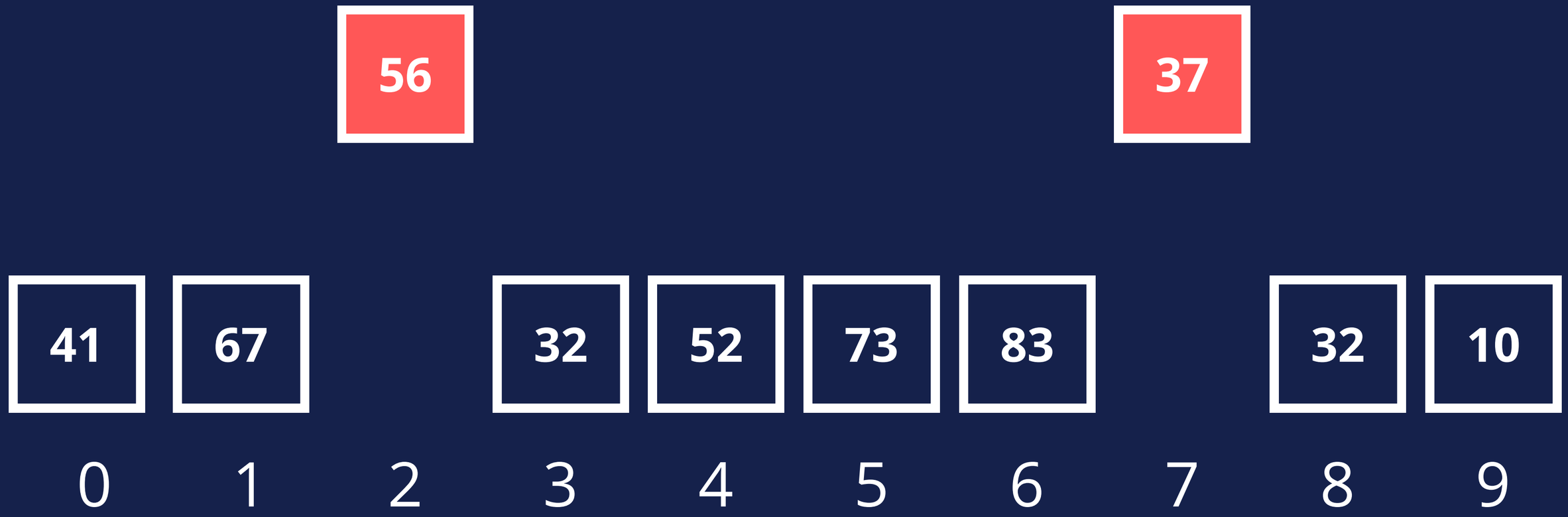
swaps = 1



Shell Sort

interval = 5

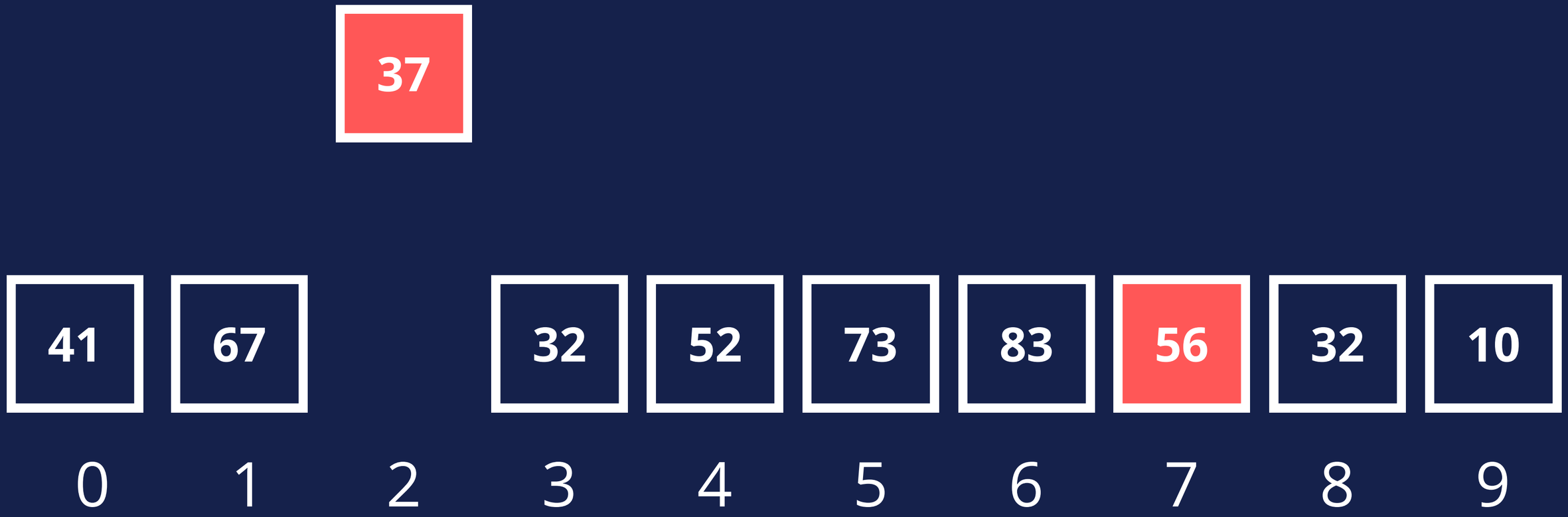
swaps = 1



Shell Sort

interval = 5

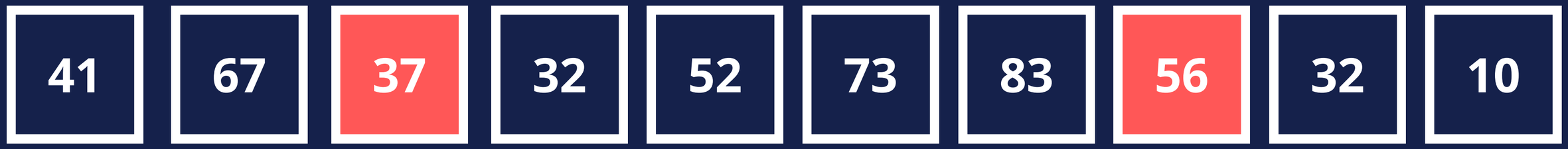
swaps = 2



Shell Sort

interval = 5

swaps = 2

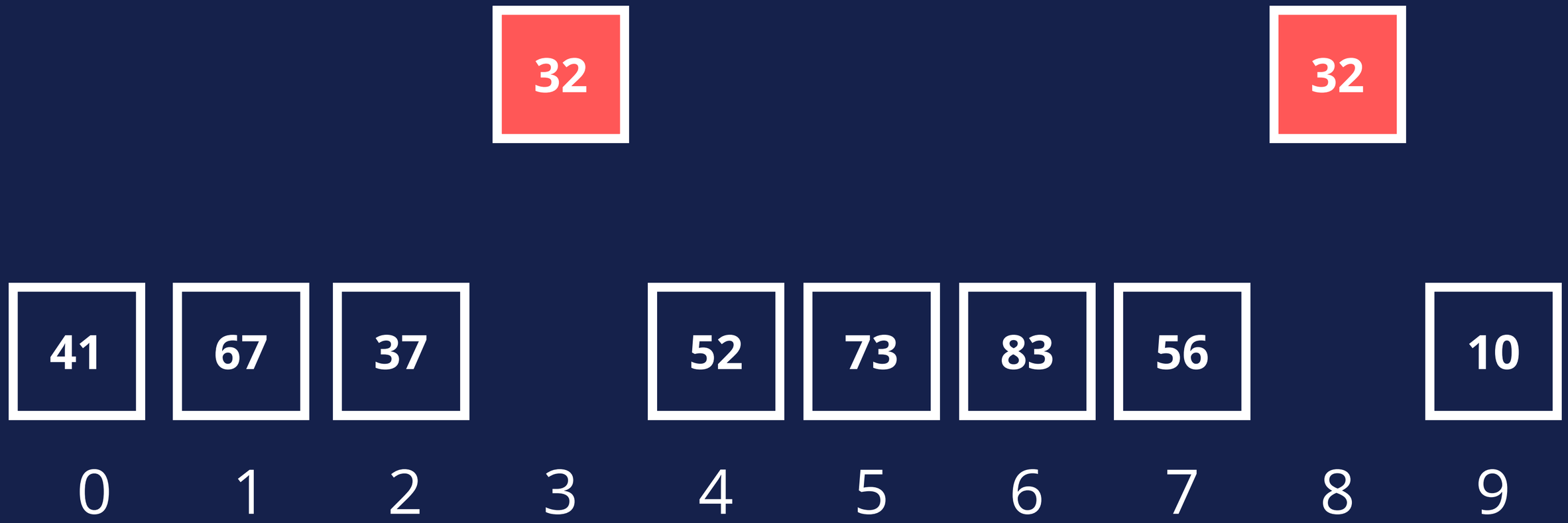


0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 5

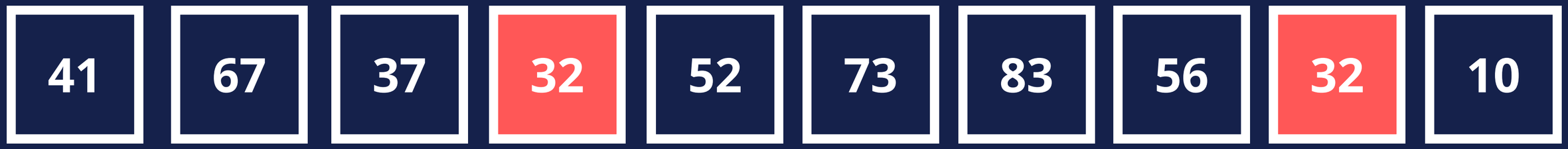
swaps = 2



Shell Sort

interval = 5

swaps = 2

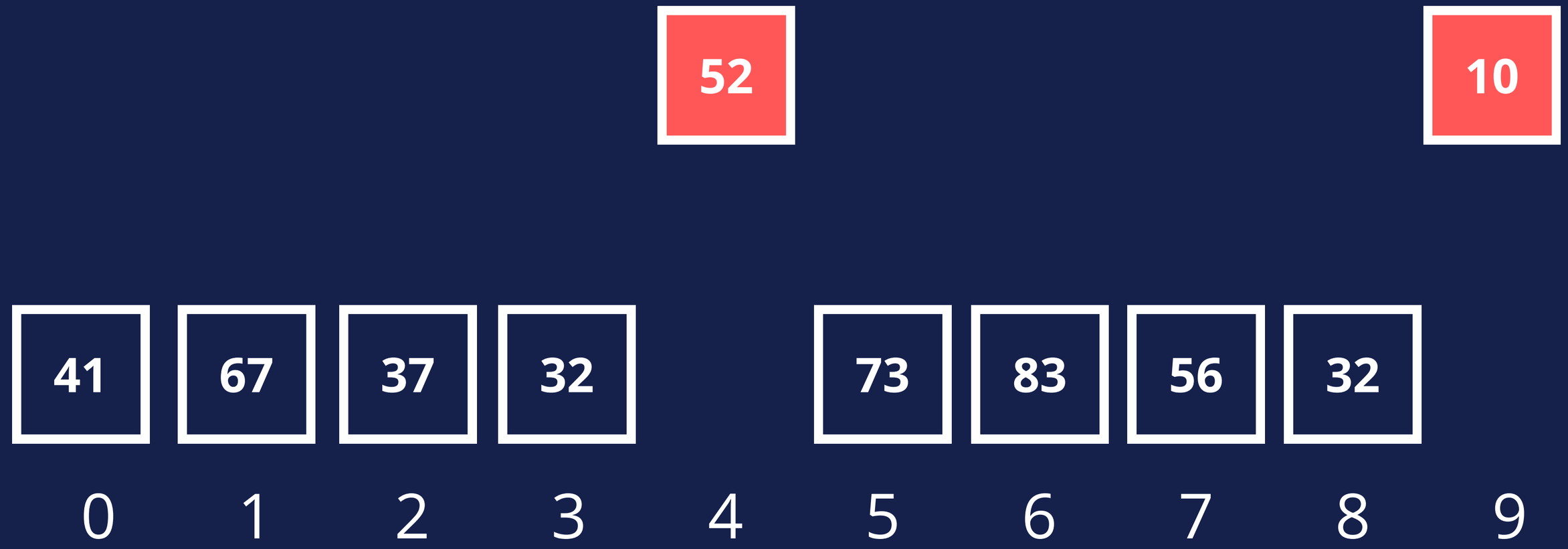


0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 5

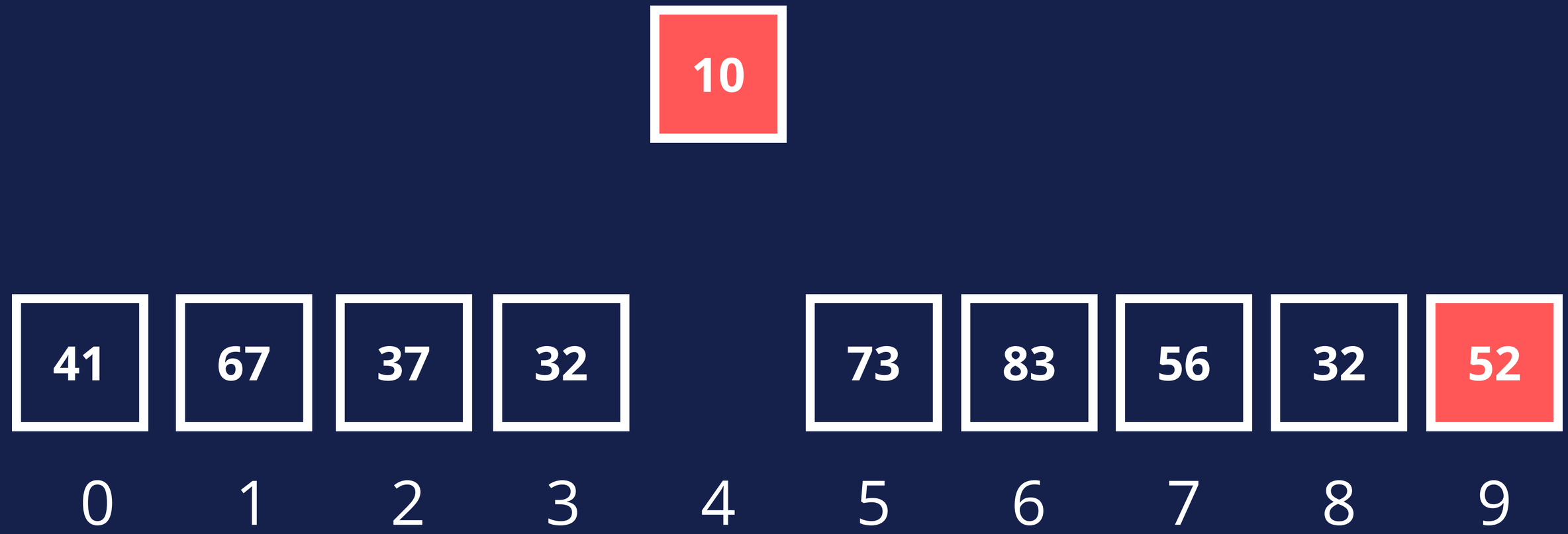
swaps = 2



Shell Sort

interval = 5

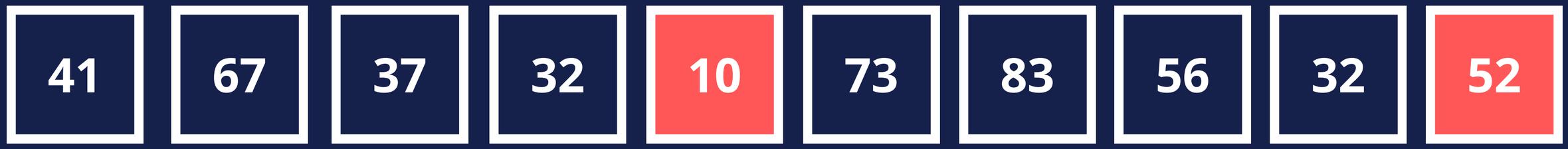
swaps = 3



Shell Sort

interval = 5

swaps = 3



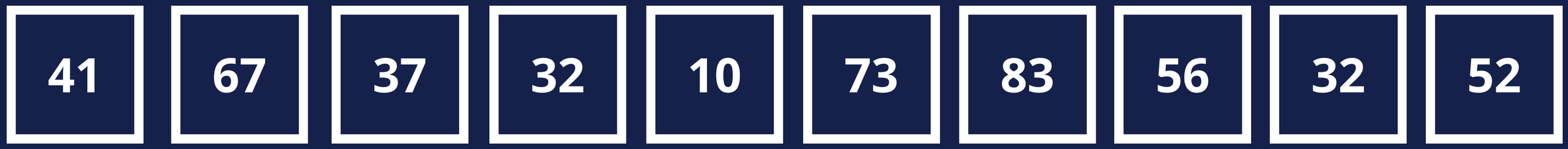
0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 5

swaps = 3

interval = interval / 2



Shell Sort

interval = 2

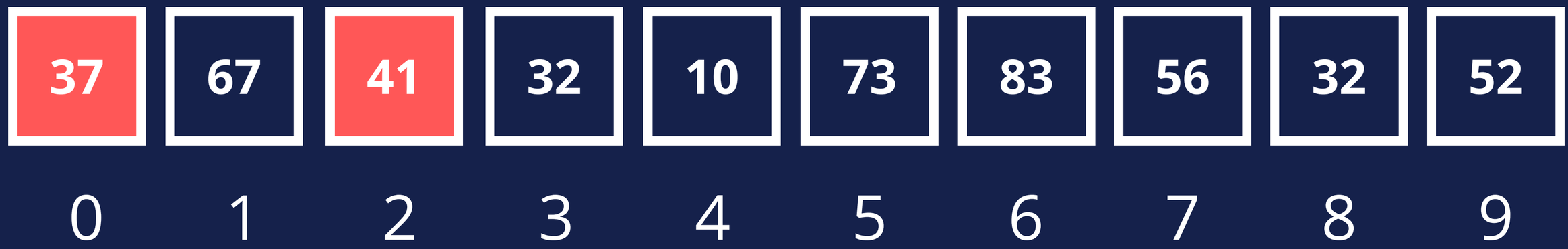
swaps = 3

Shell Sort

interval = 2

swaps = 4

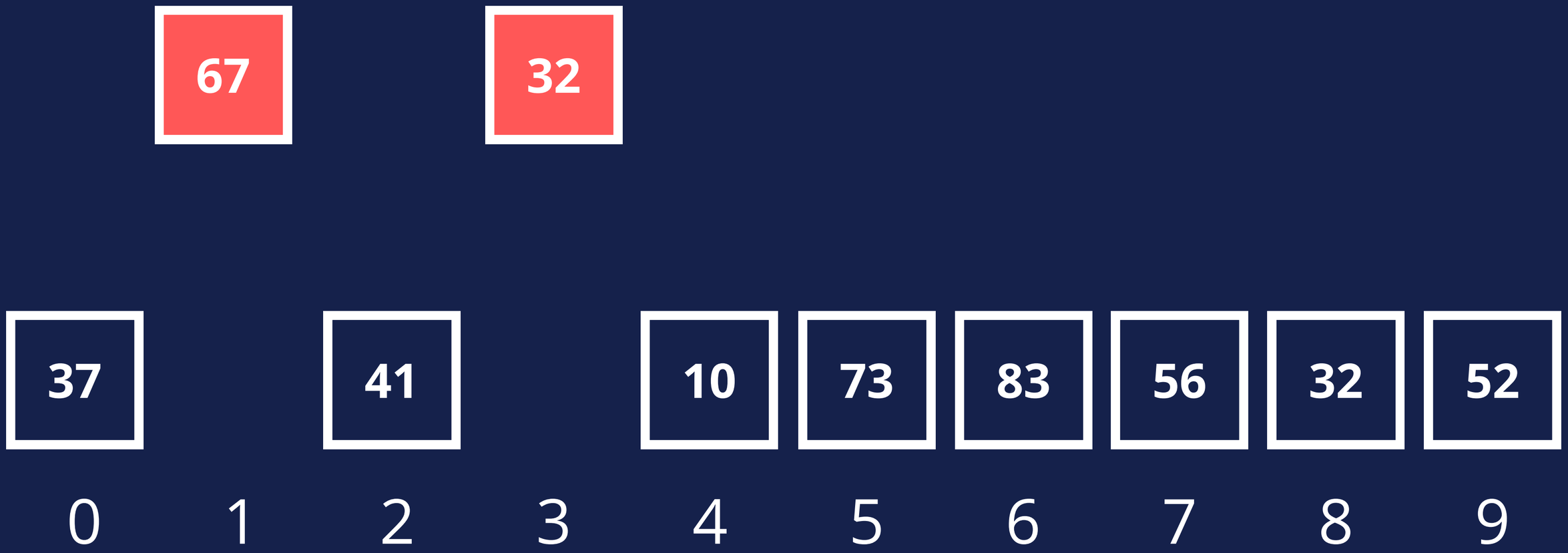
Shell Sort interval = 2 swaps = 4



Shell Sort

interval = 2

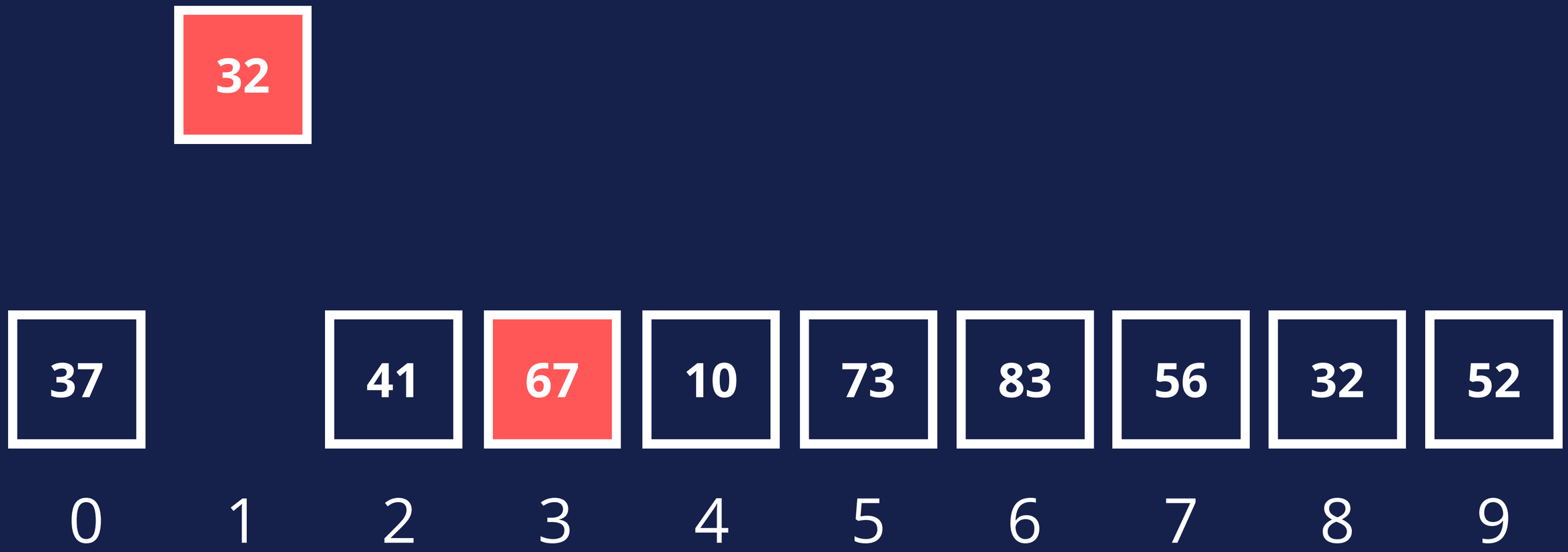
swaps = 4



Shell Sort

interval = 2

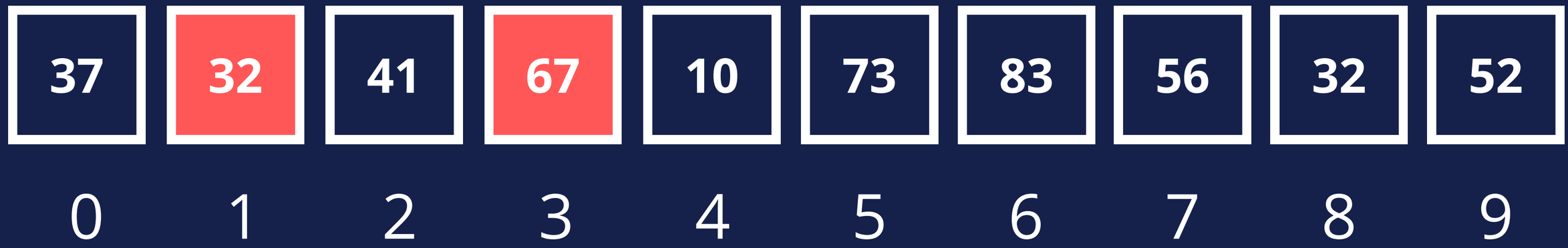
swaps = 5



Shell Sort

interval = 2

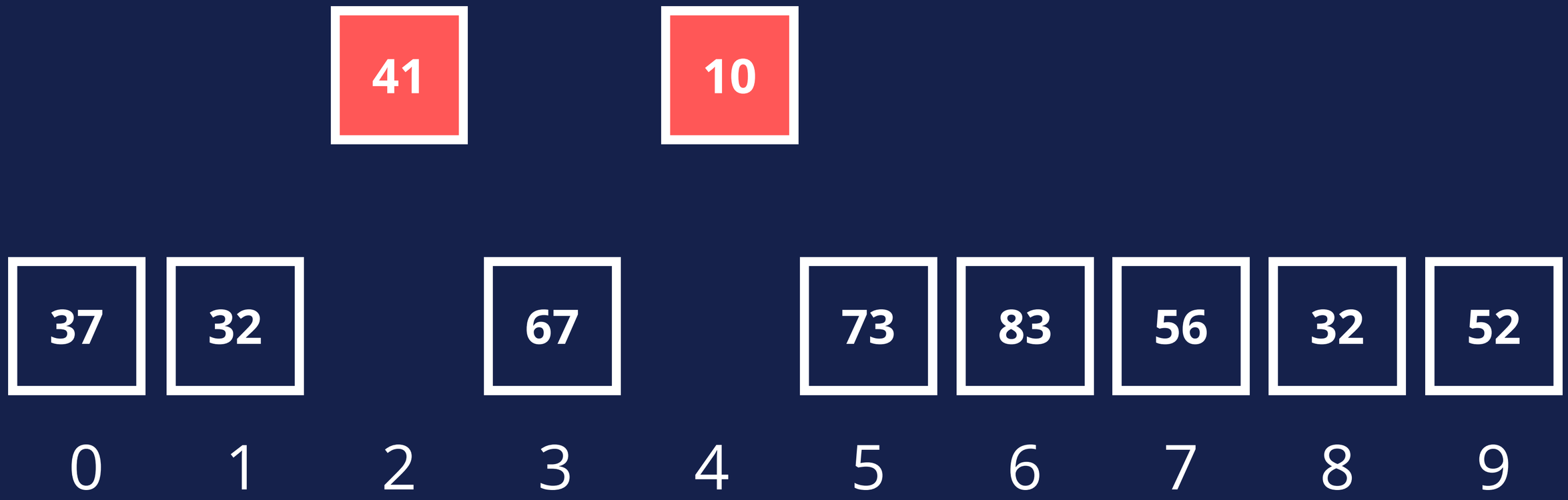
swaps = 5



Shell Sort

interval = 2

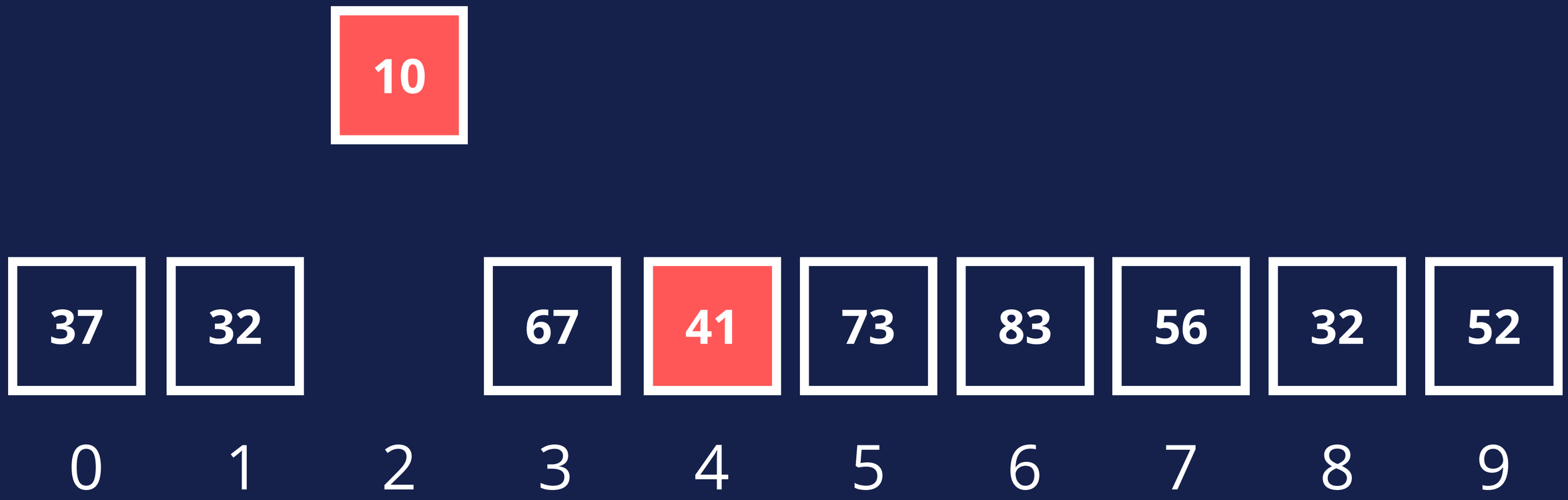
swaps = 5



Shell Sort

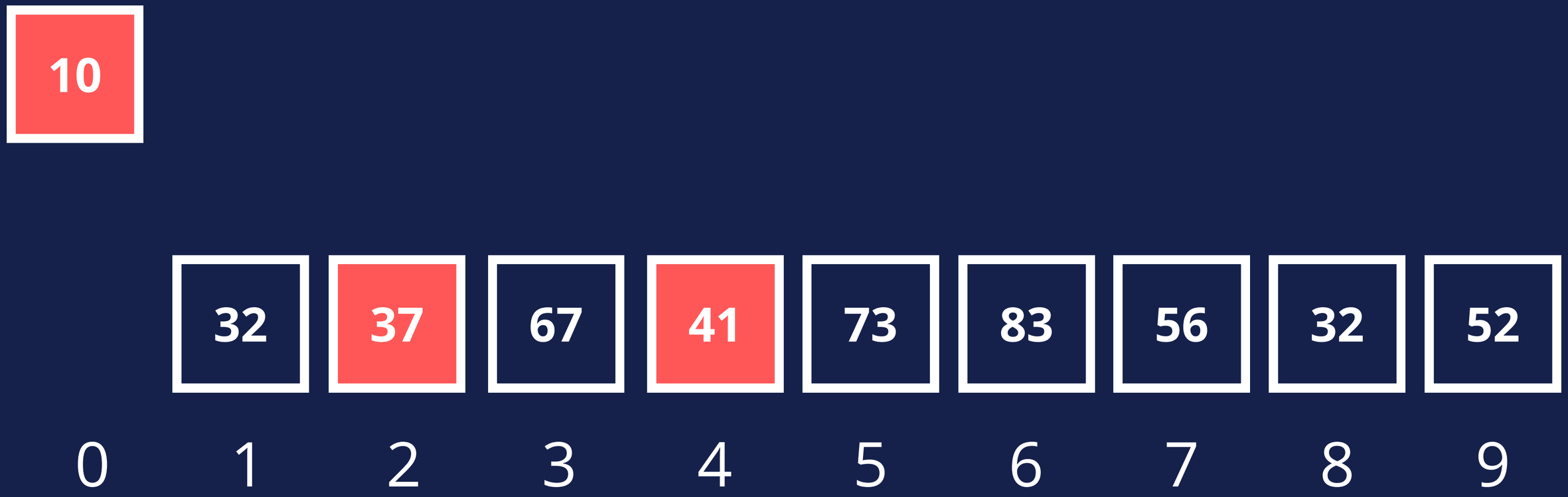
interval = 2

swaps = 6

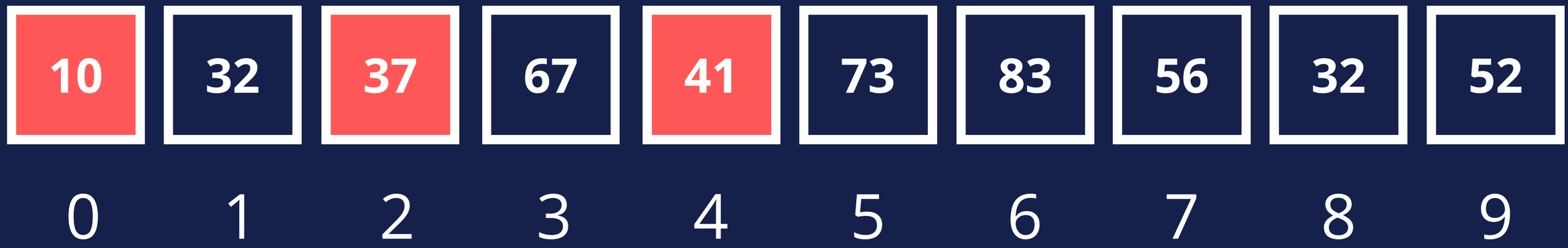


Shell Sort interval = 2 swaps = 6

Shell Sort interval = 2 swaps = 7



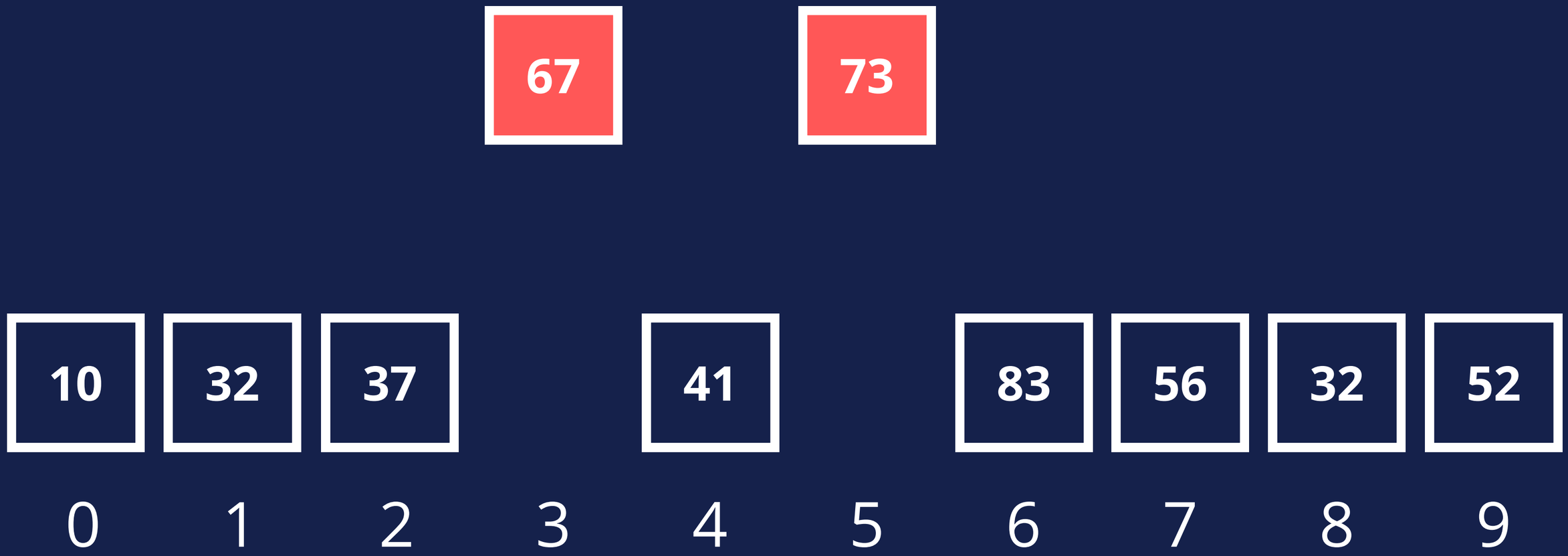
Shell Sort interval = 2 swaps = 7



Shell Sort

interval = 2

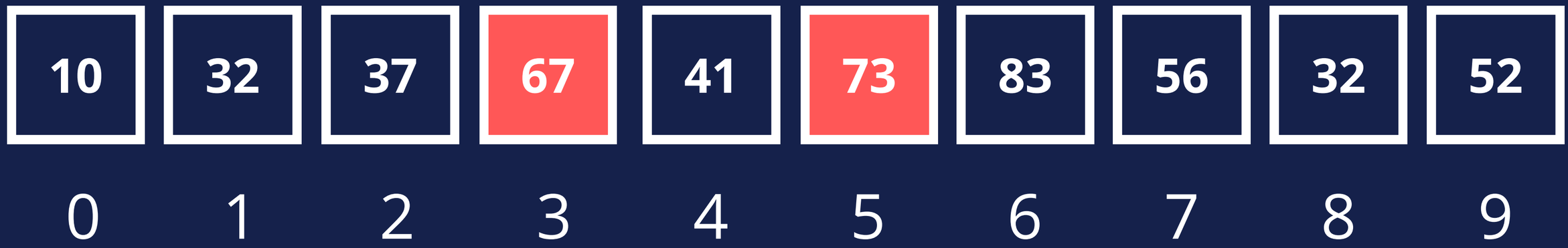
swaps = 7



Shell Sort

interval = 2

swaps = 7



Shell Sort

interval = 2

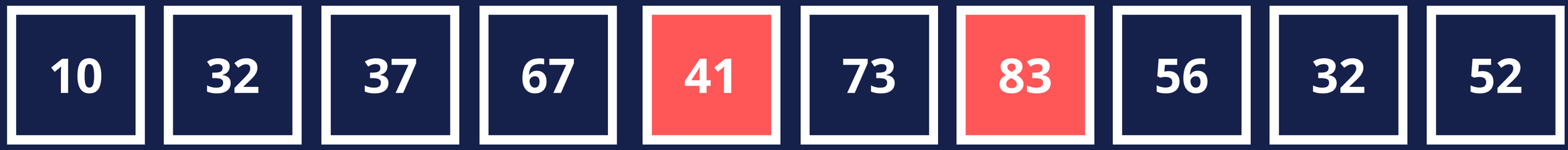
swaps = 7

0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 2

swaps = 7

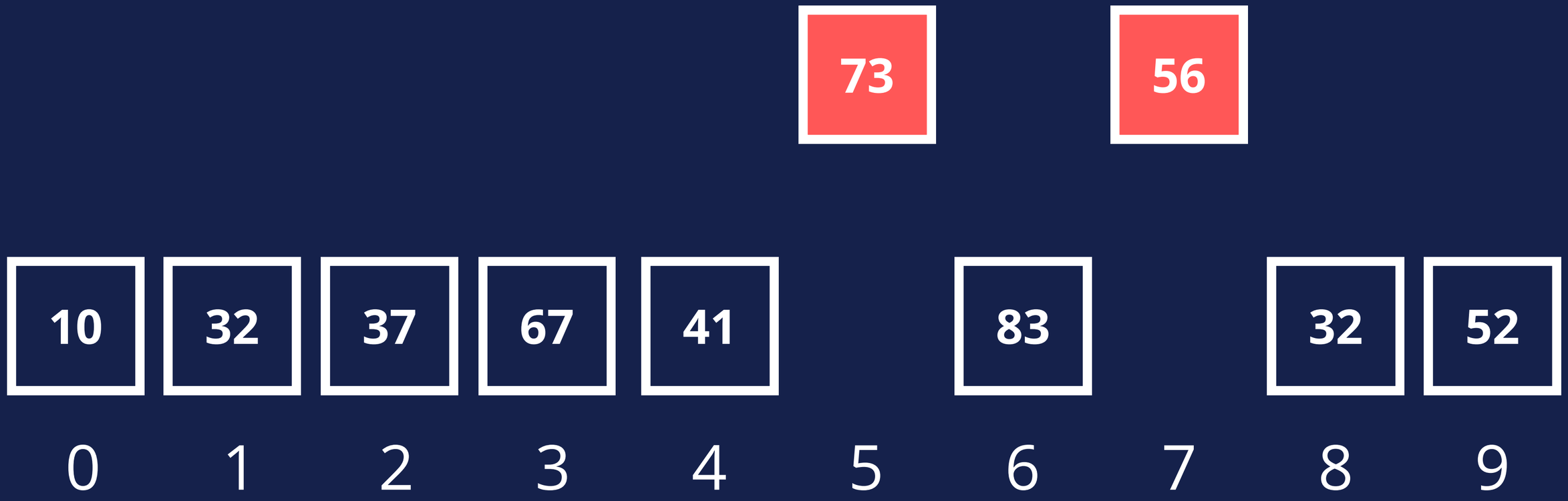


0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 2

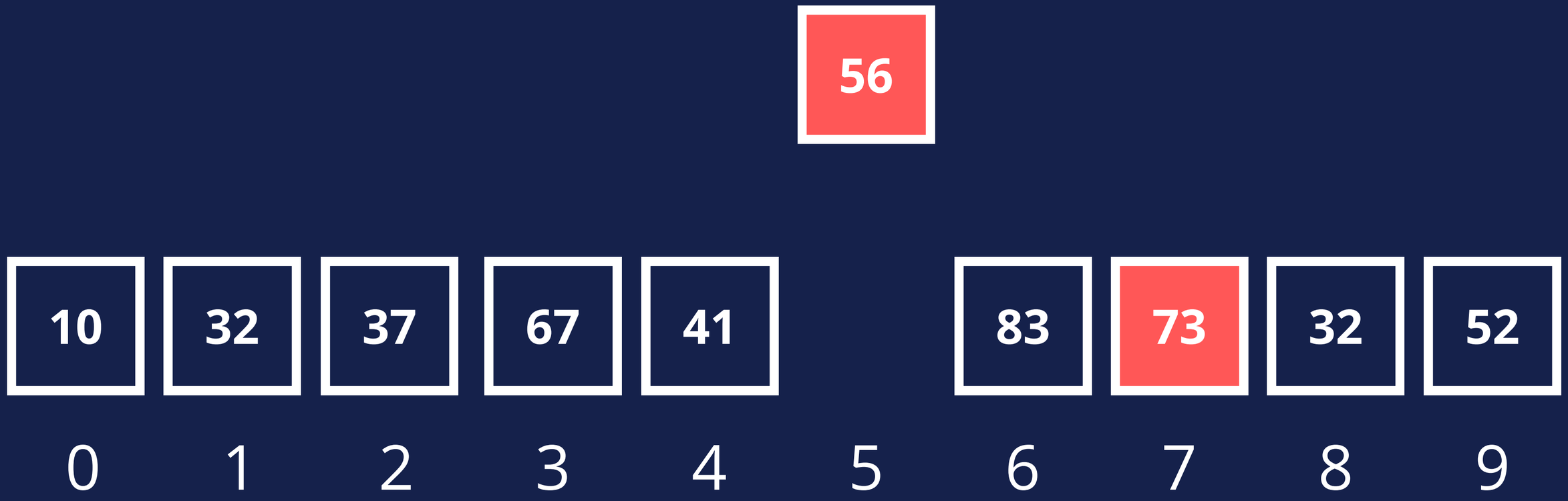
swaps = 7



Shell Sort

interval = 2

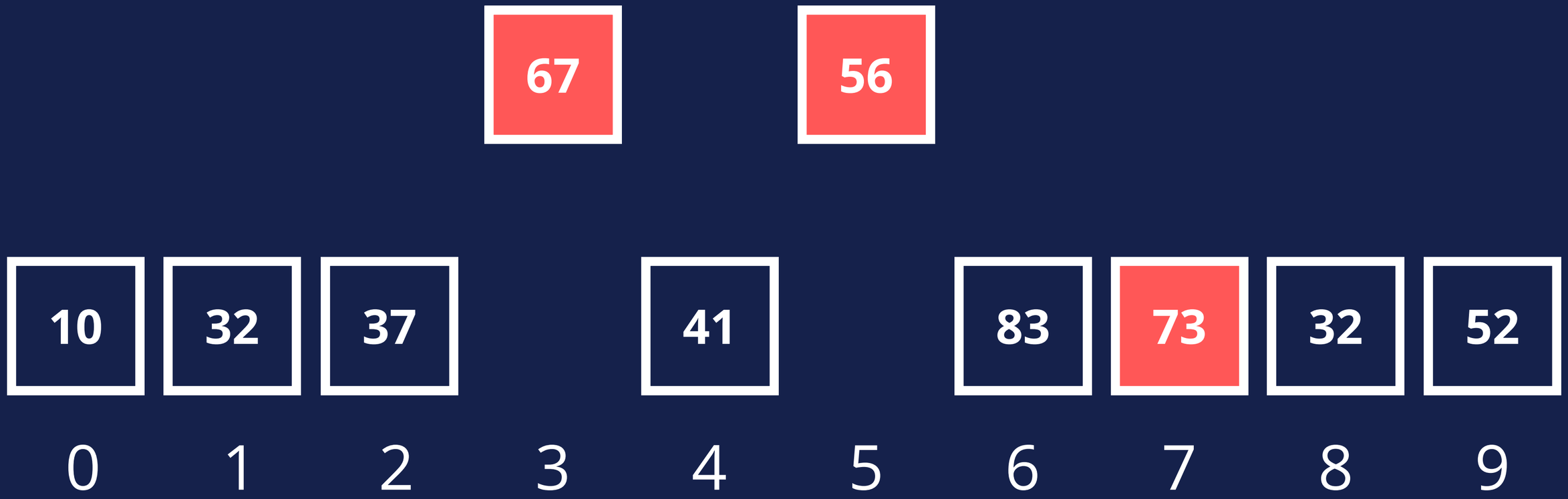
swaps = 8



Shell Sort

interval = 2

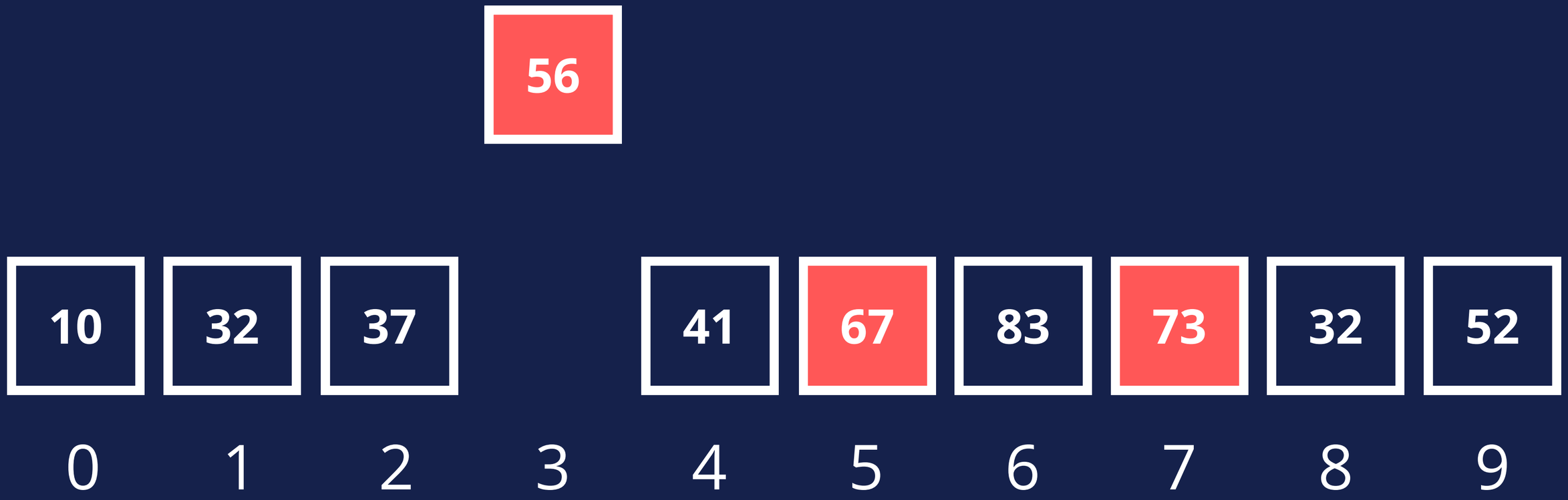
swaps = 8



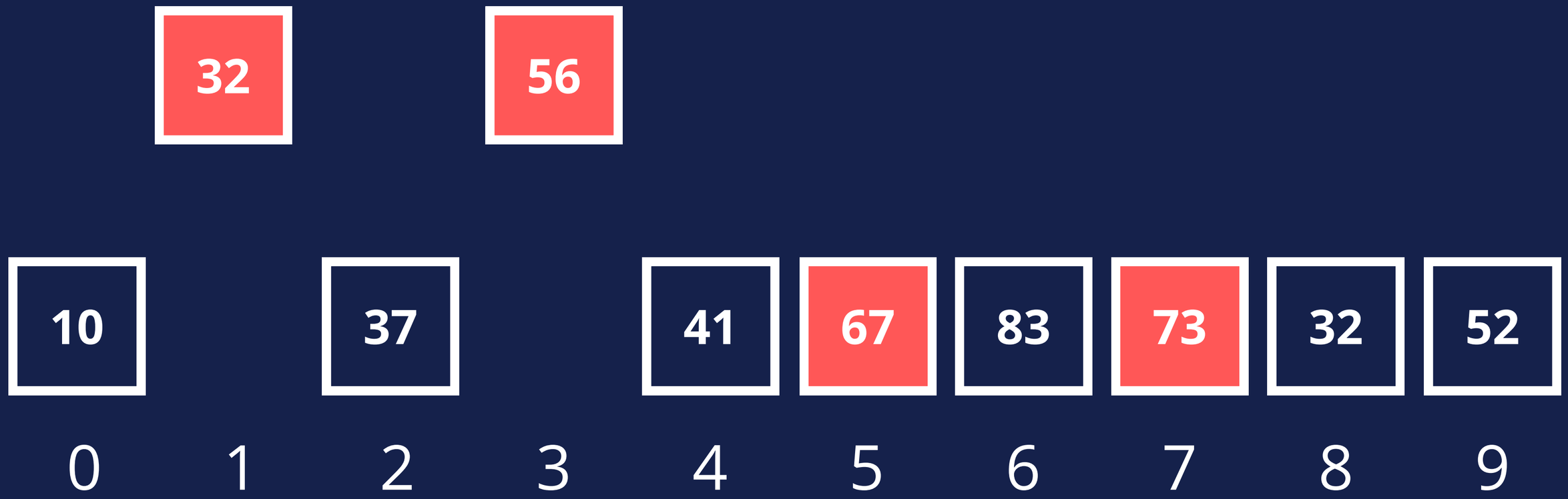
Shell Sort

interval = 2

swaps = 9



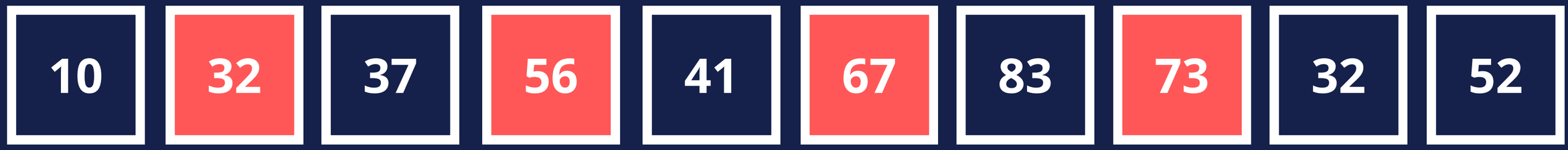
Shell Sort interval = 2 swaps = 9



Shell Sort

interval = 2

swaps = 9

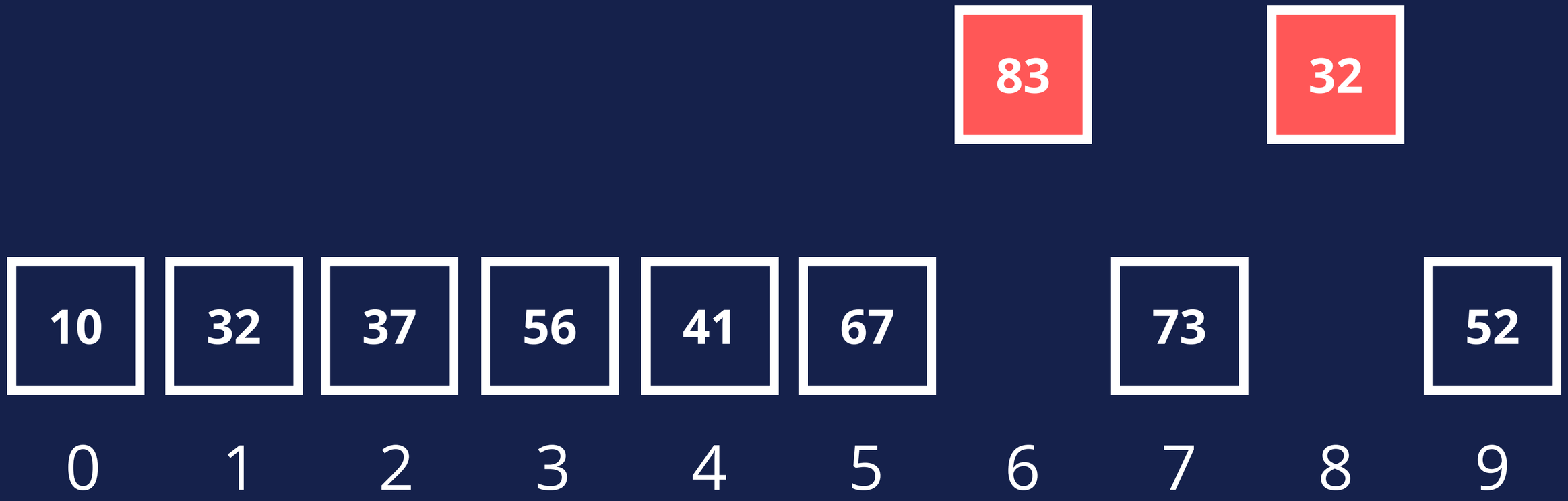


0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 2

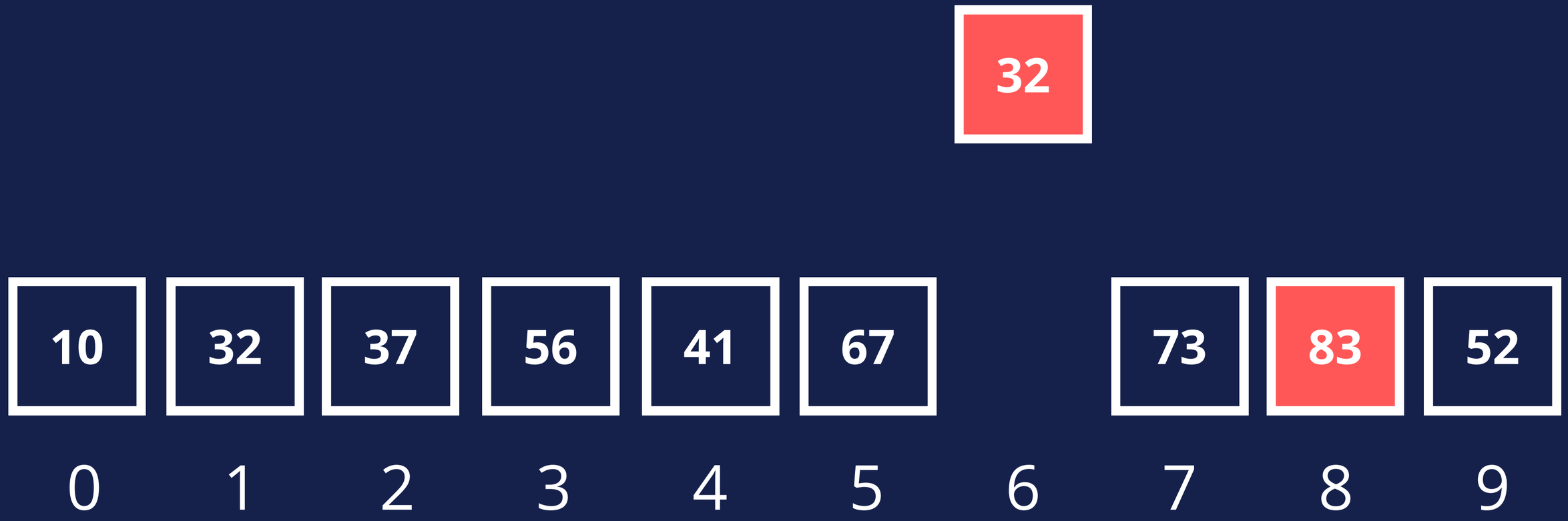
swaps = 9



Shell Sort

interval = 2

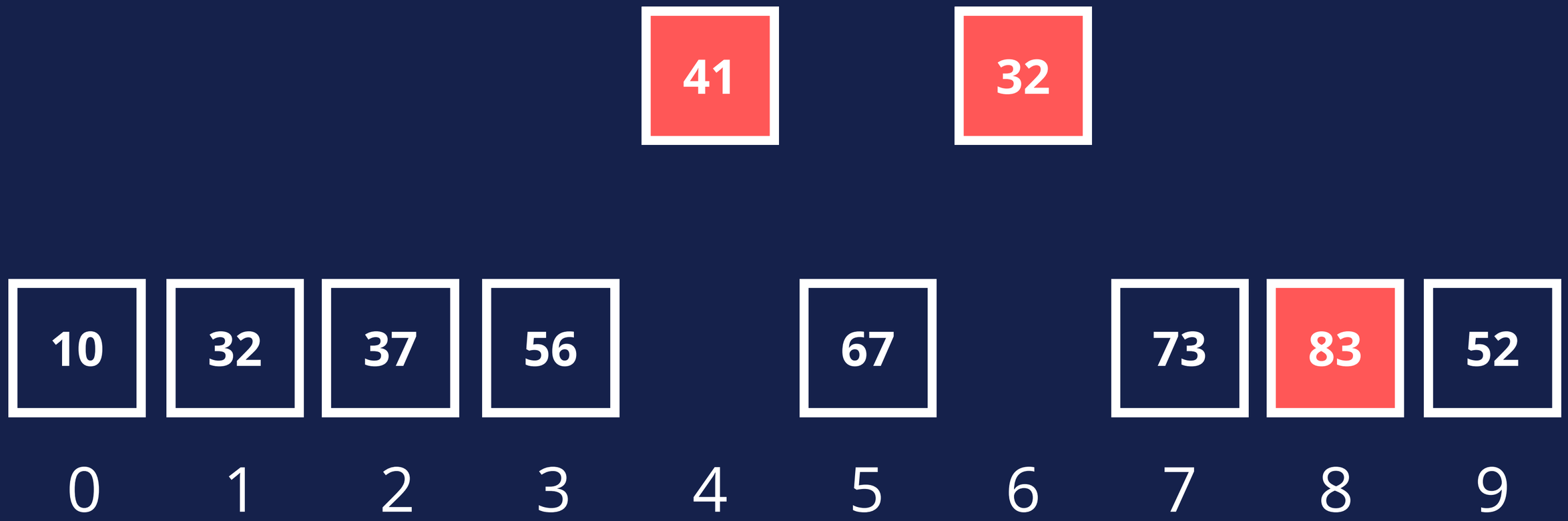
swaps = 10



Shell Sort

interval = 2

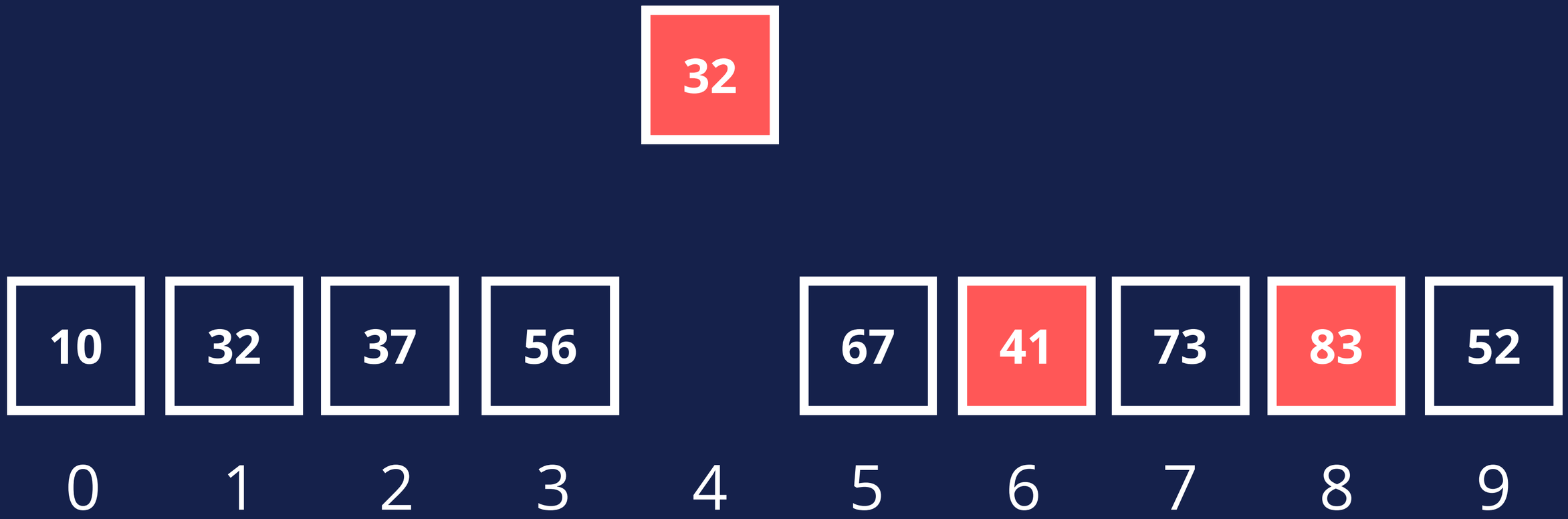
swaps = 10



Shell Sort

interval = 2

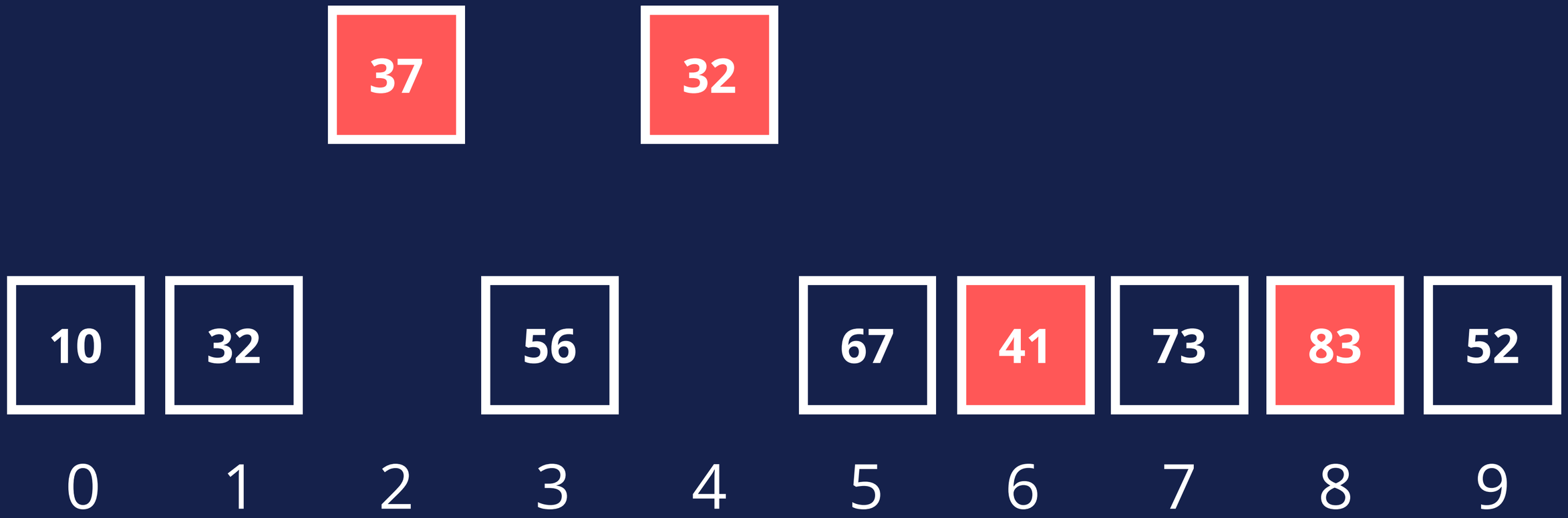
swaps = 11



Shell Sort

interval = 2

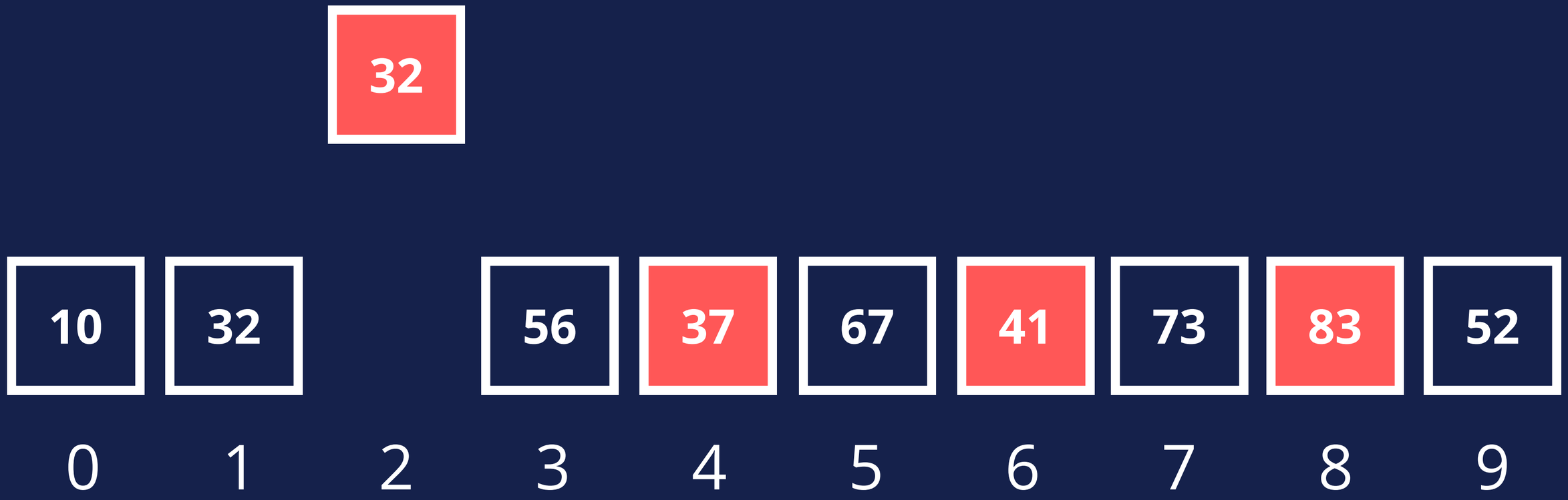
swaps = 11



Shell Sort

interval = 2

swaps = 12

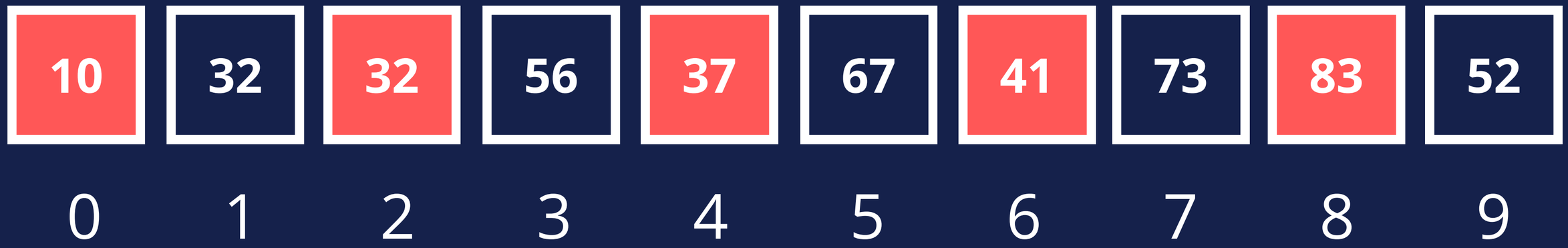


Shell Sort interval = 2 swaps = 12

Shell Sort

interval = 2

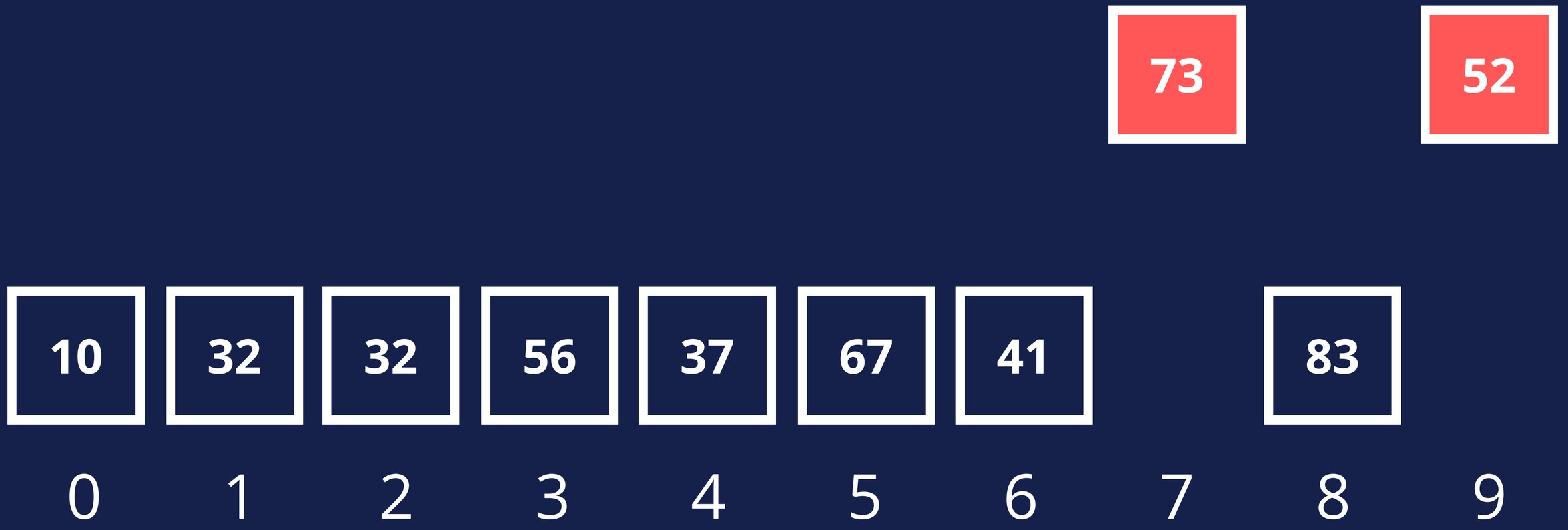
swaps = 12



Shell Sort

interval = 2

swaps = 12



Shell Sort

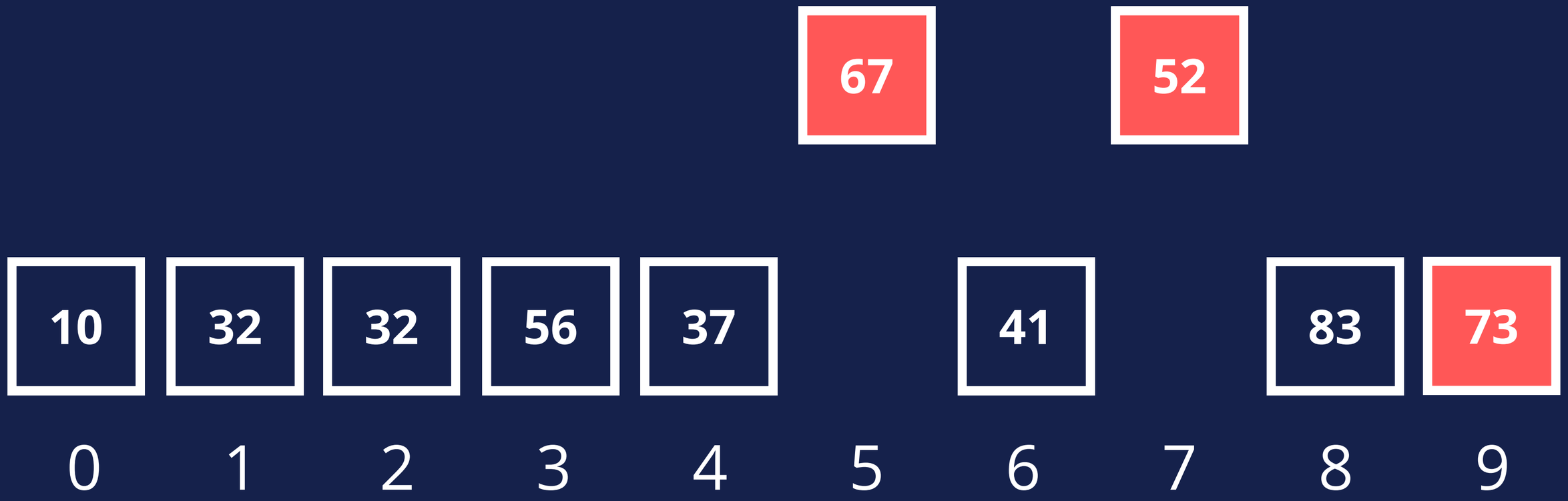
interval = 2

swaps = 13

Shell Sort

interval = 2

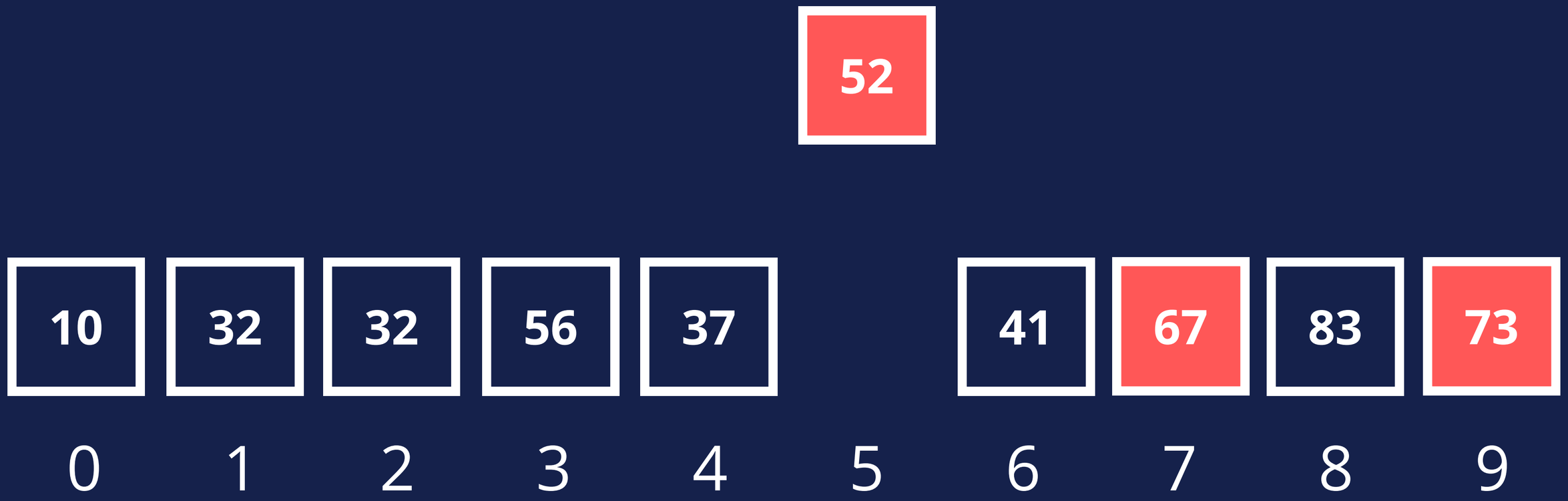
swaps = 13



Shell Sort

interval = 2

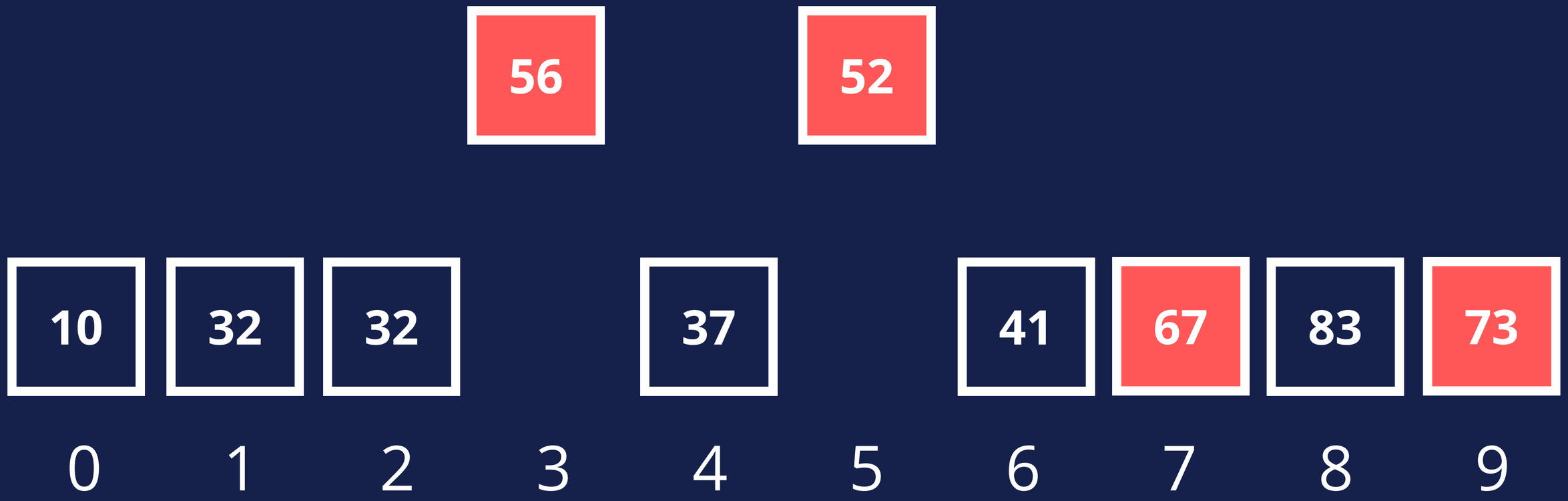
swaps = 14



Shell Sort

interval = 2

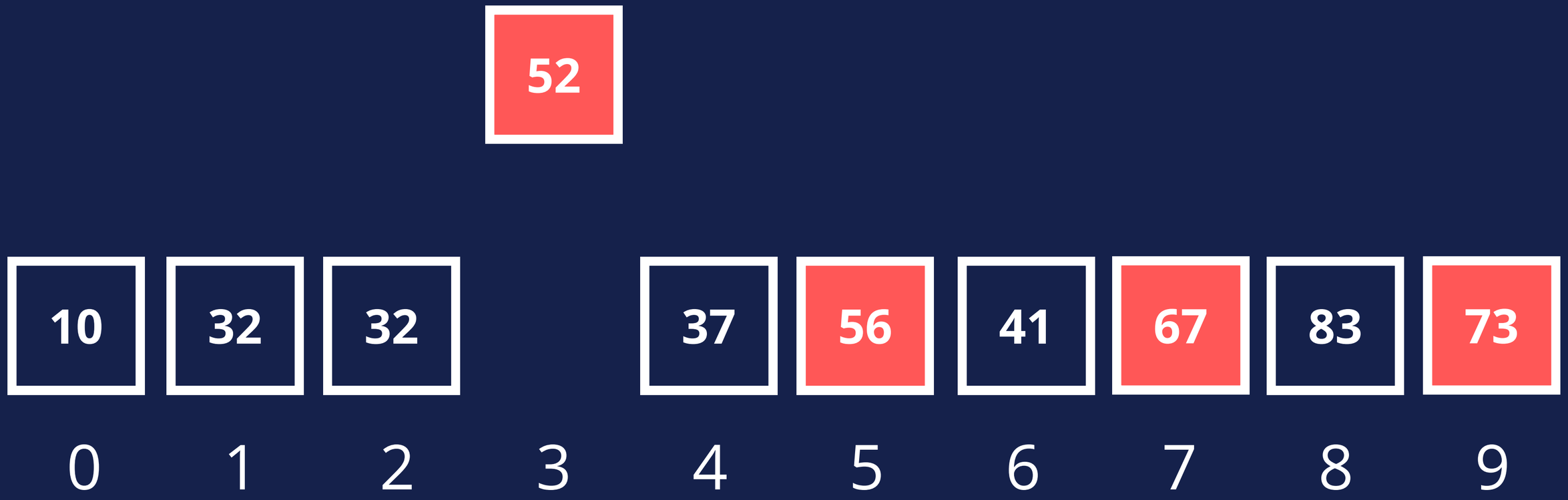
swaps = 14



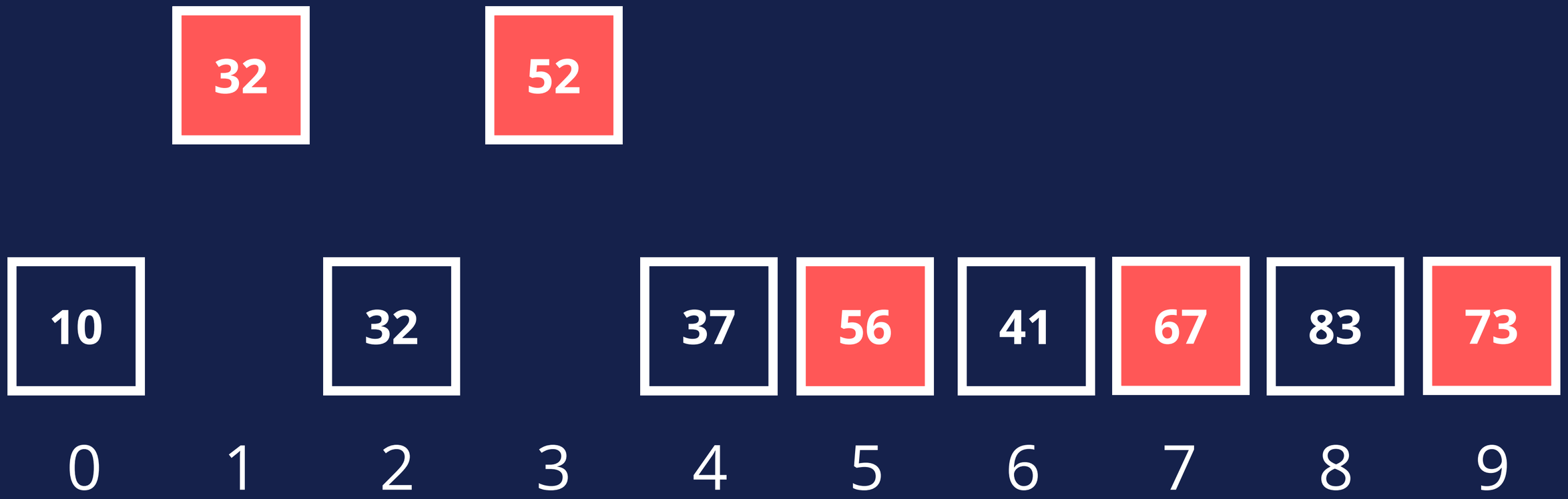
Shell Sort

interval = 2

swaps = 15



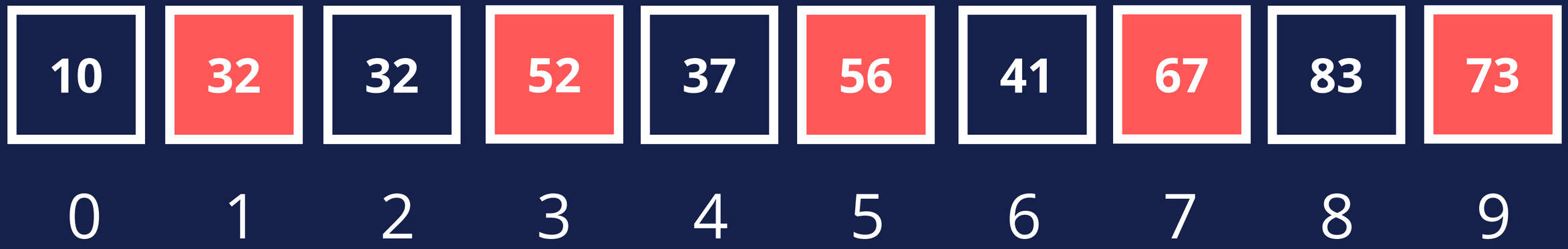
Shell Sort interval = 2 swaps = 15



Shell Sort

interval = 2

swaps = 15



Shell Sort

interval = 2

swaps = 15

interval = interval / 2

Shell Sort interval = 1 swaps = 15

Shell Sort

interval = 1

swaps = 15

Shell Sort

interval = 1

swaps = 15

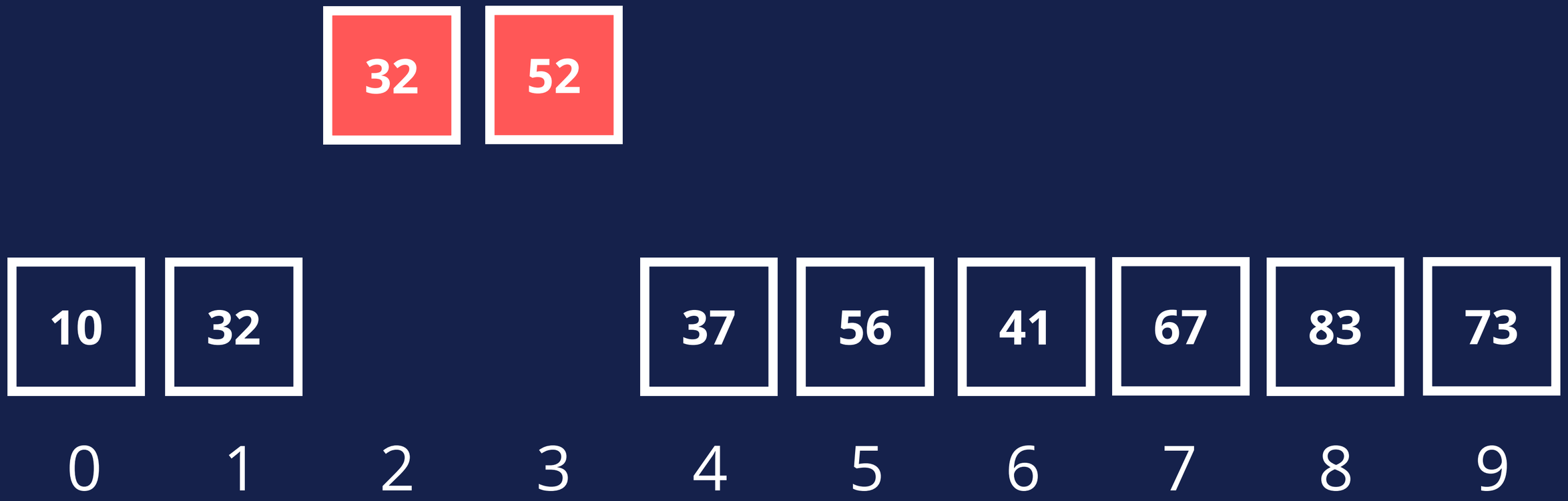
Shell Sort interval = 1 swaps = 15

Shell Sort interval = 1 swaps = 15

Shell Sort

interval = 1

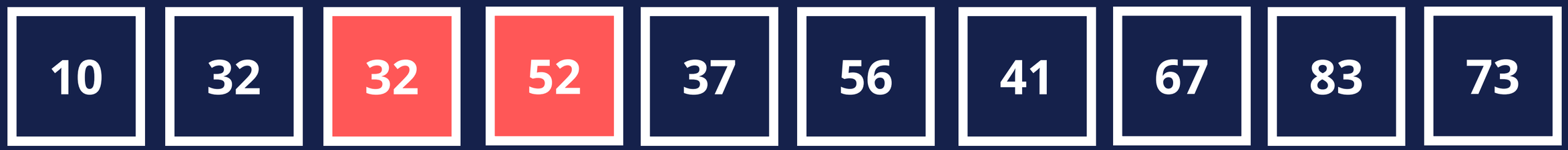
swaps = 15



Shell Sort

interval = 1

swaps = 15

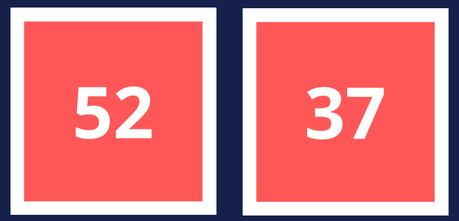


0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 1

swaps = 15

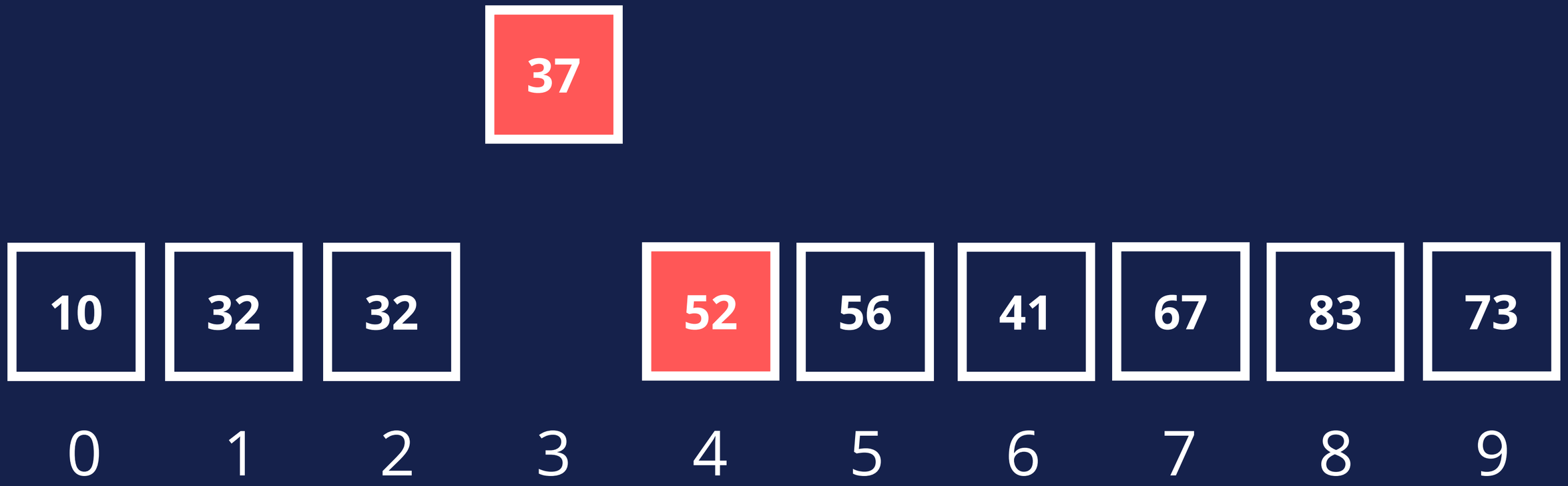


0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 1

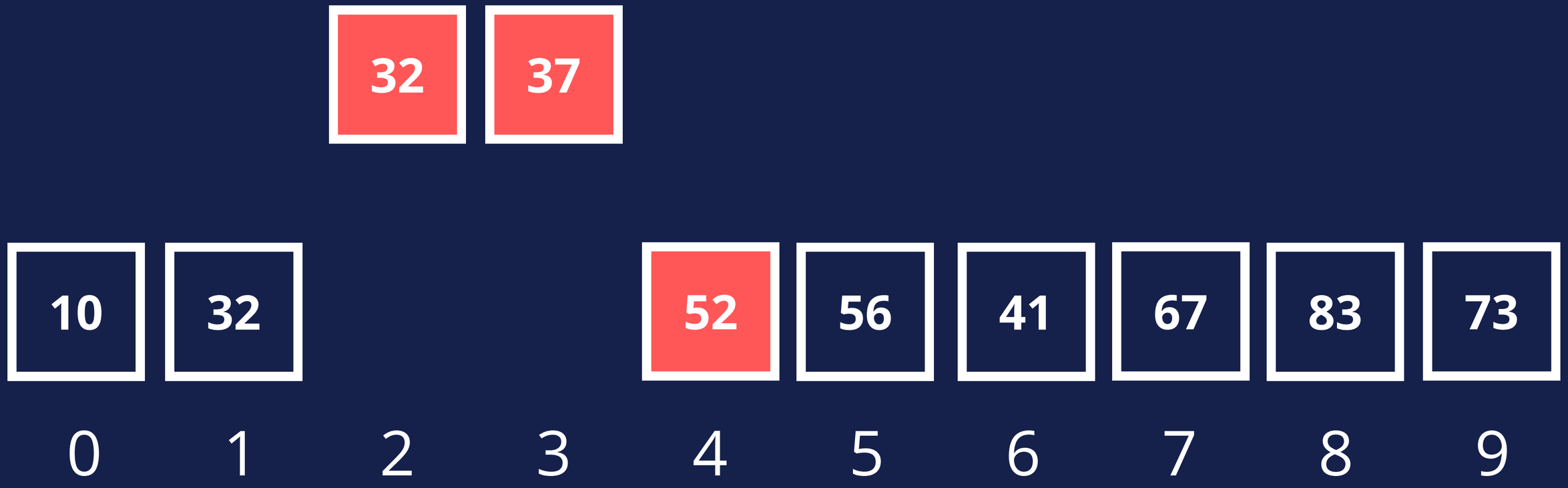
swaps = 16



Shell Sort

interval = 1

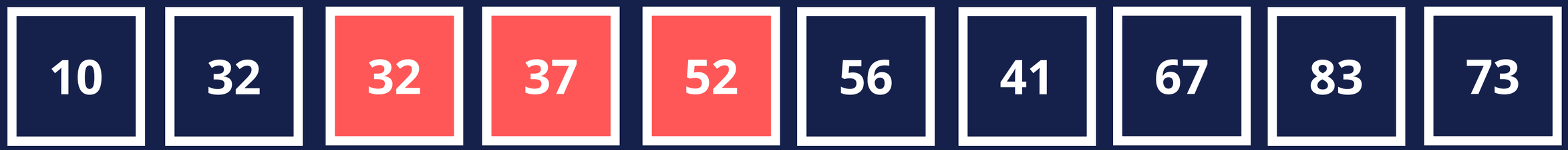
swaps = 16



Shell Sort

interval = 1

swaps = 16

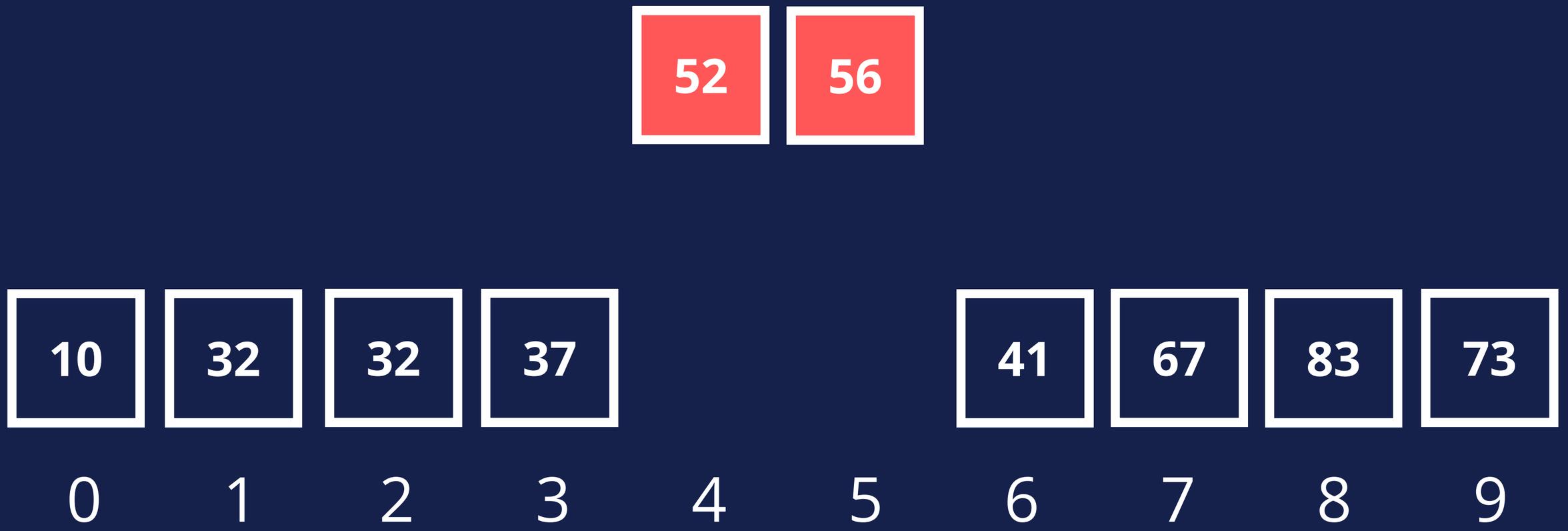


0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 1

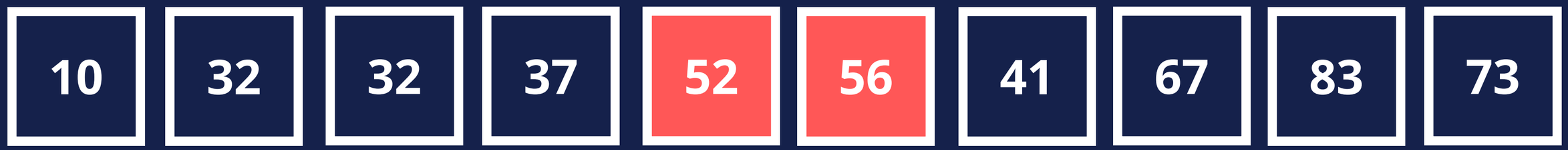
swaps = 16



Shell Sort

interval = 1

swaps = 16

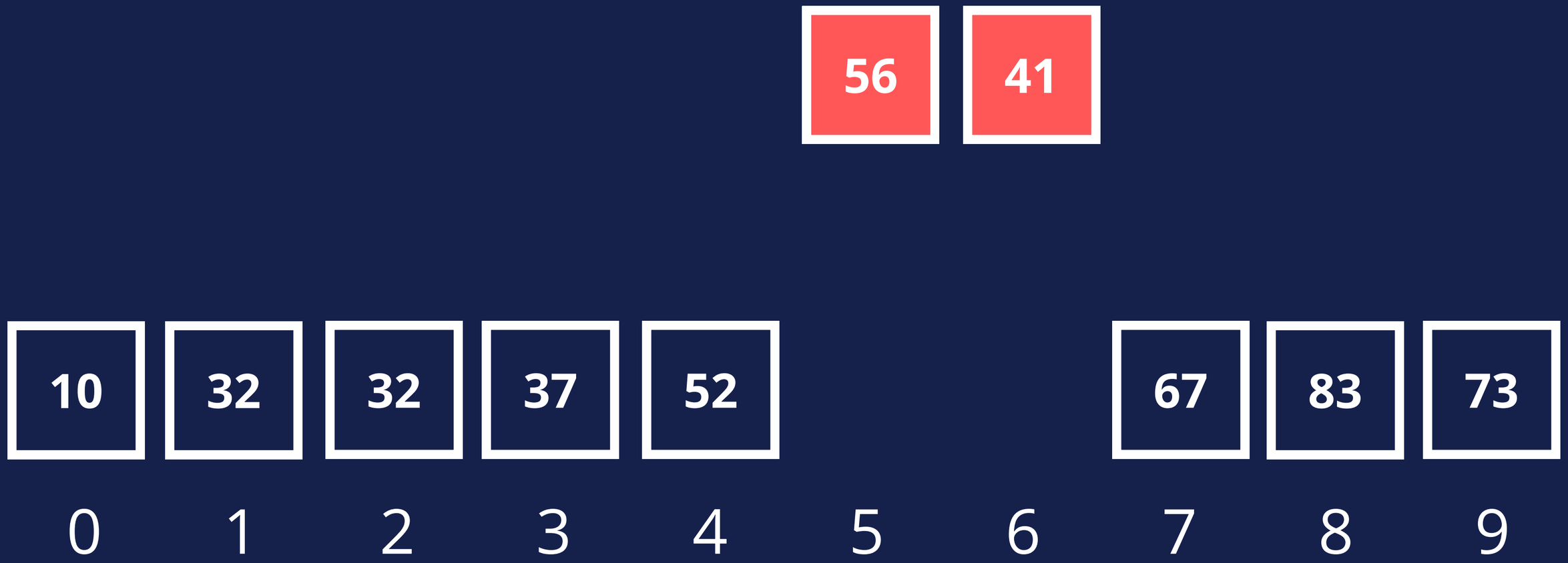


0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 1

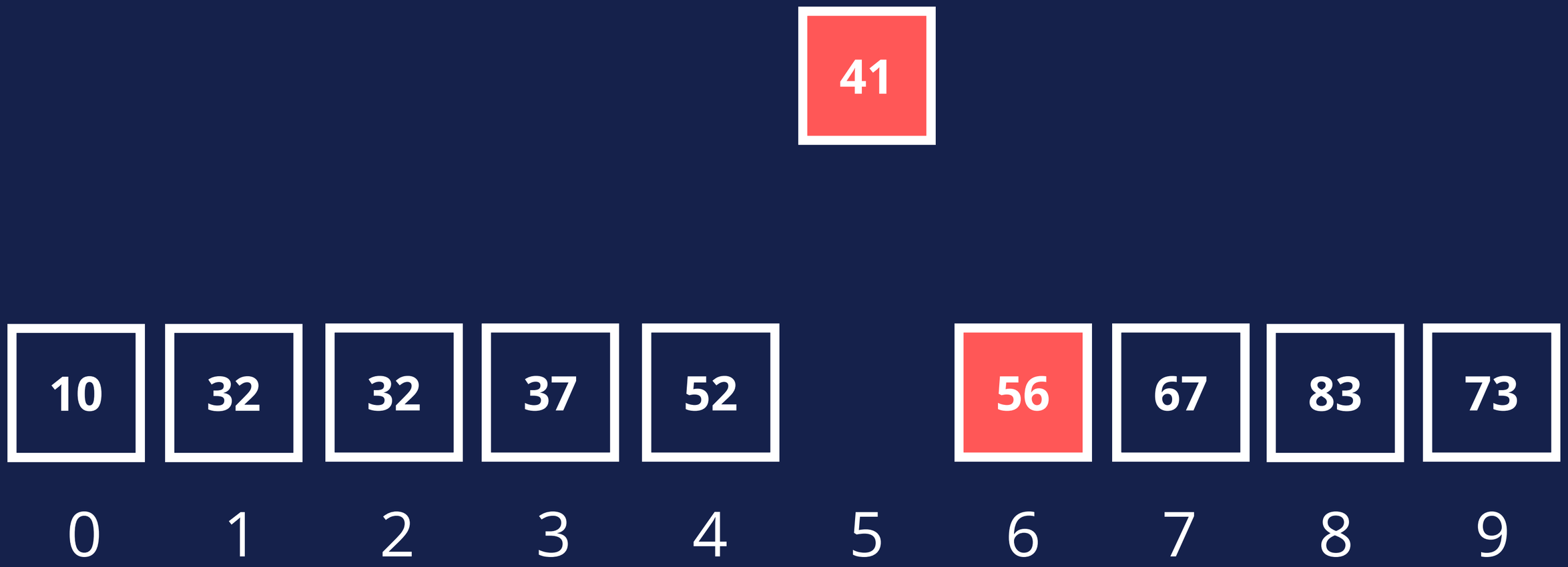
swaps = 16



Shell Sort

interval = 1

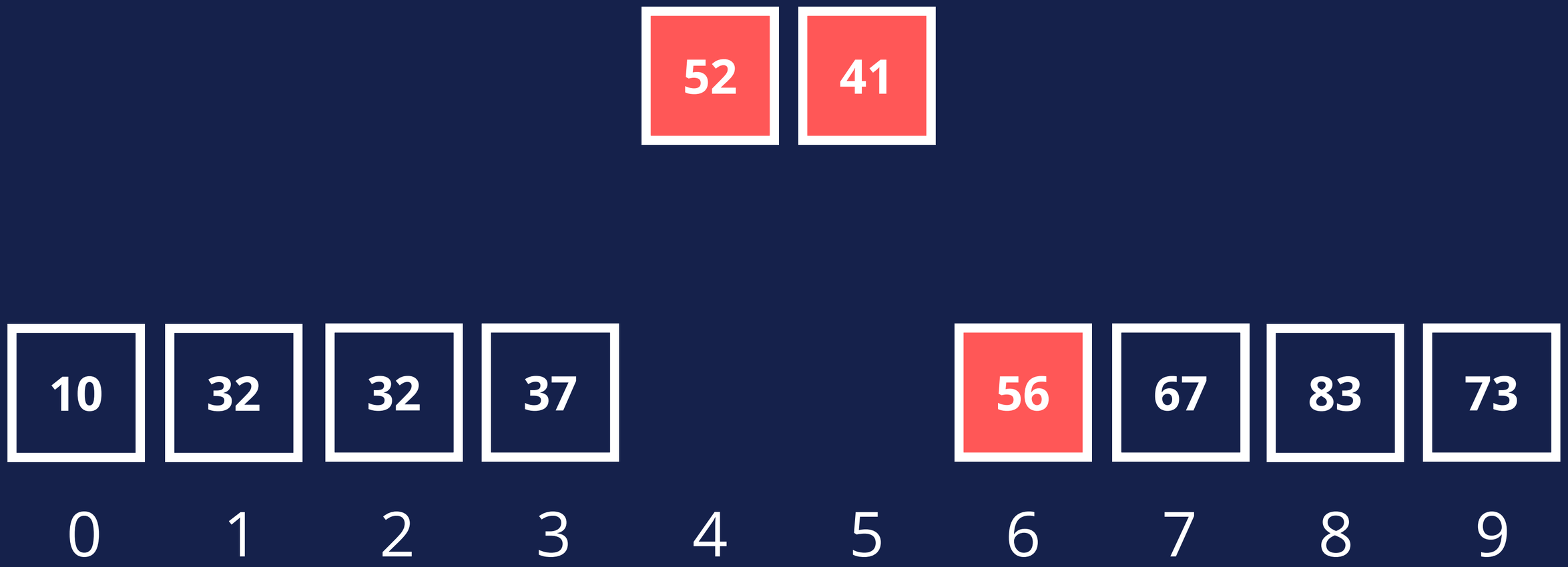
swaps = 17



Shell Sort

interval = 1

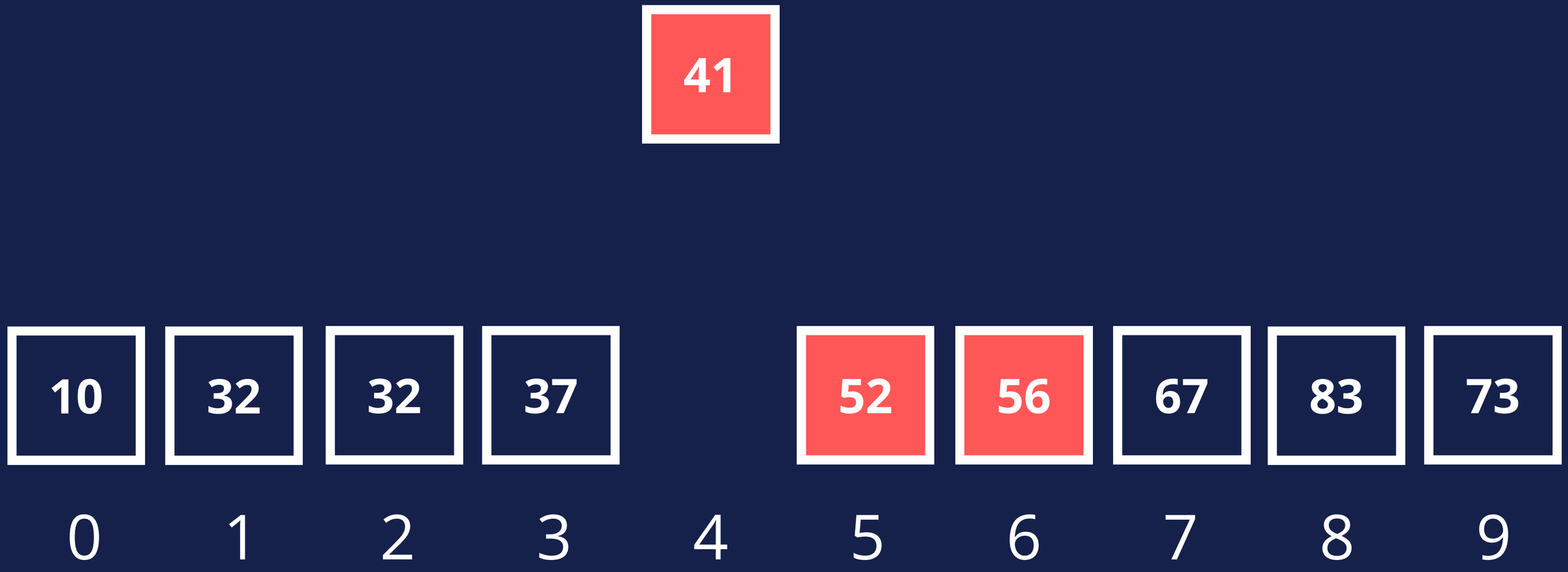
swaps = 17



Shell Sort

interval = 1

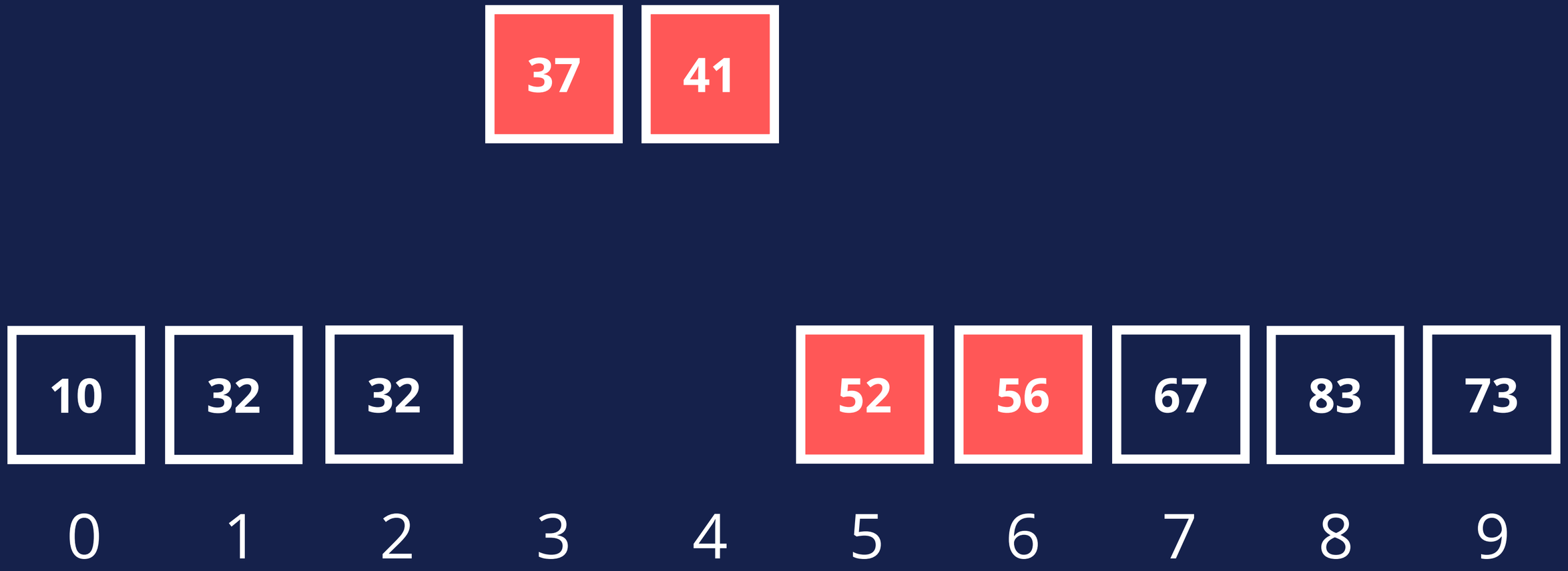
swaps = 18



Shell Sort

interval = 1

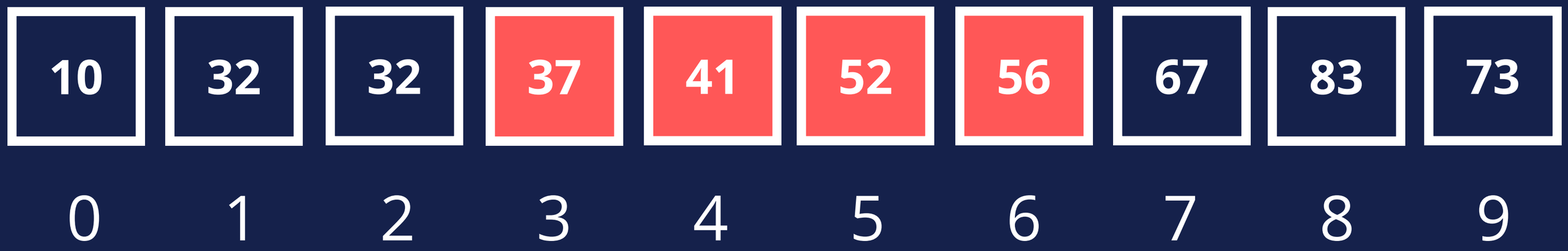
swaps = 18



Shell Sort

interval = 1

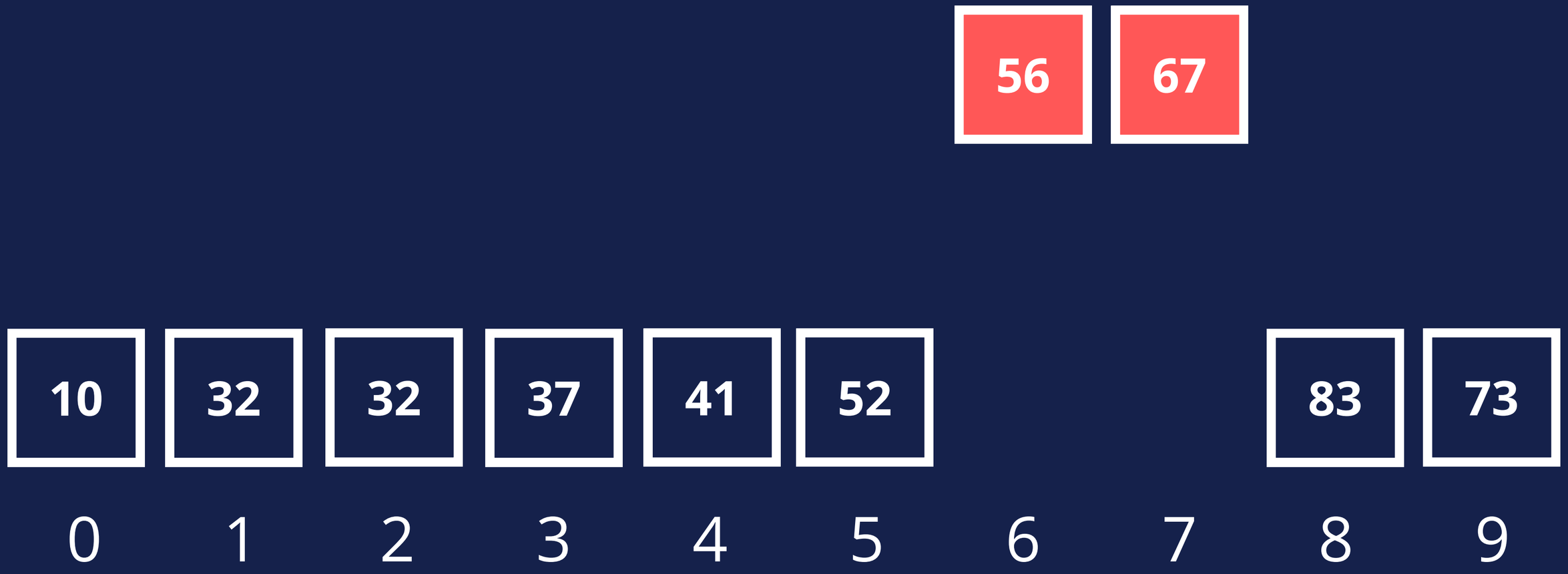
swaps = 18



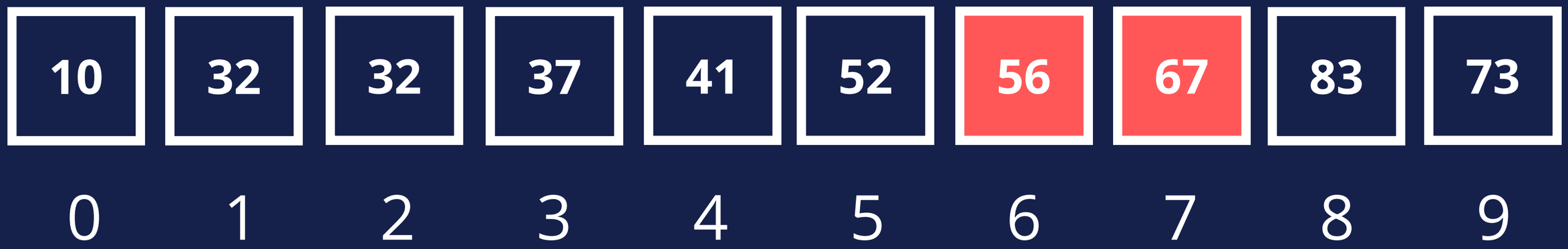
Shell Sort

interval = 1

swaps = 18



Shell Sort interval = 1 swaps = 18

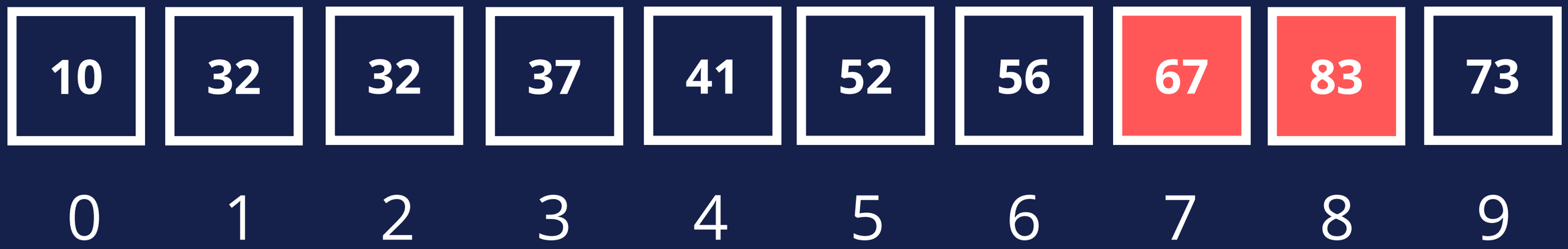


Shell Sort

interval = 1

swaps = 18

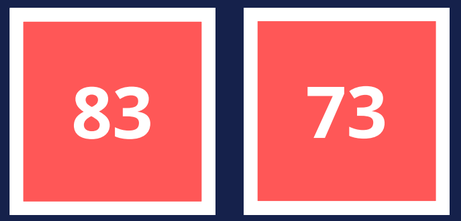
Shell Sort interval = 1 swaps = 18

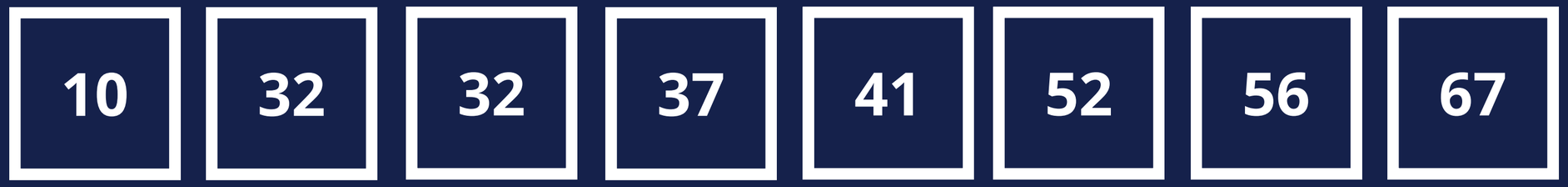


Shell Sort

interval = 1

swaps = 18





0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 1

swaps = 19

Shell Sort

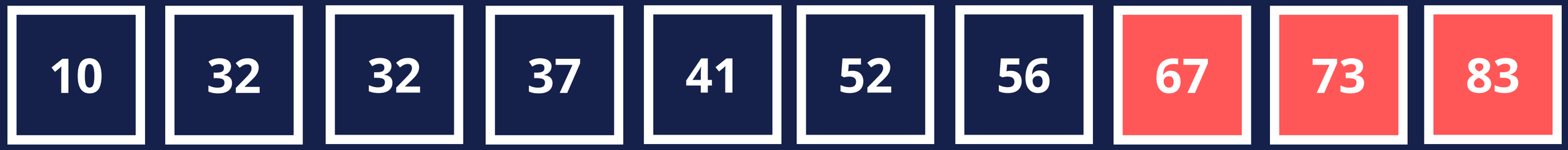
interval = 1

swaps = 19

Shell Sort

interval = 1

swaps = 19



0 1 2 3 4 5 6 7 8 9

Shell Sort

interval = 1

swaps = 19

interval = interval / 2

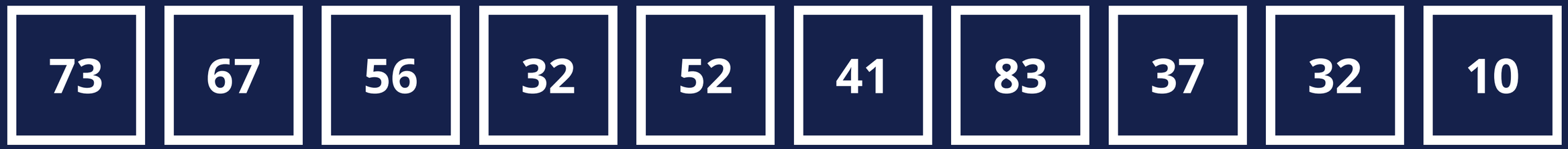
Shell Sort interval = 0 swaps = 19

Shell Sort using Knuth

interval = ?

swaps = 0

1, 4, 13, 40, 121, 364, 1093, ...



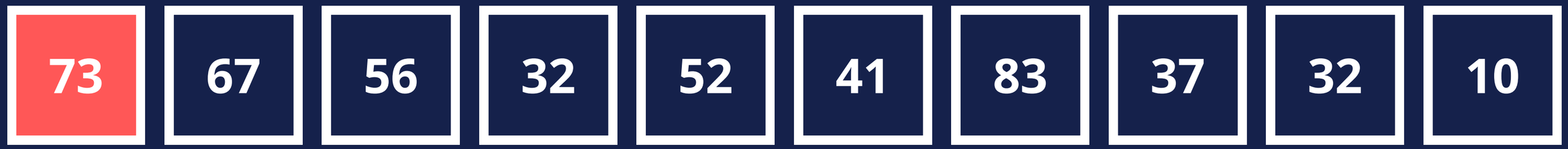
0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 4

swaps = 0

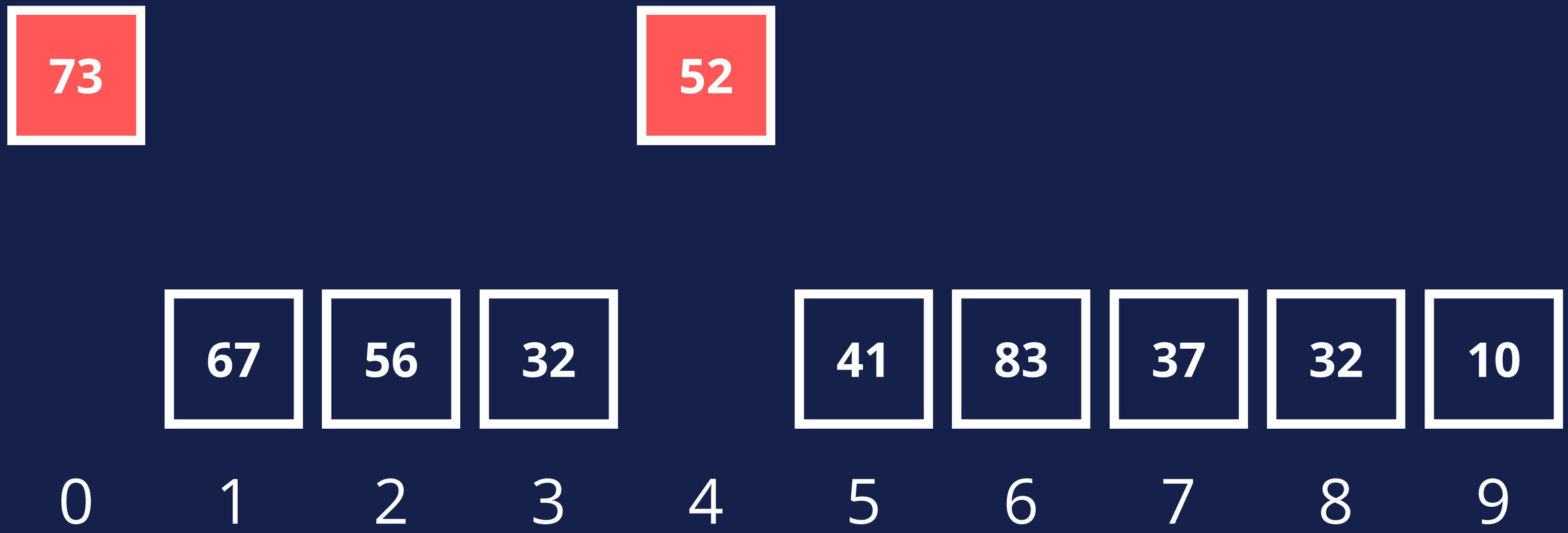
1, 4, 13, 40, 121, 364, 1093, ...



Shell Sort using Knuth

interval = 4

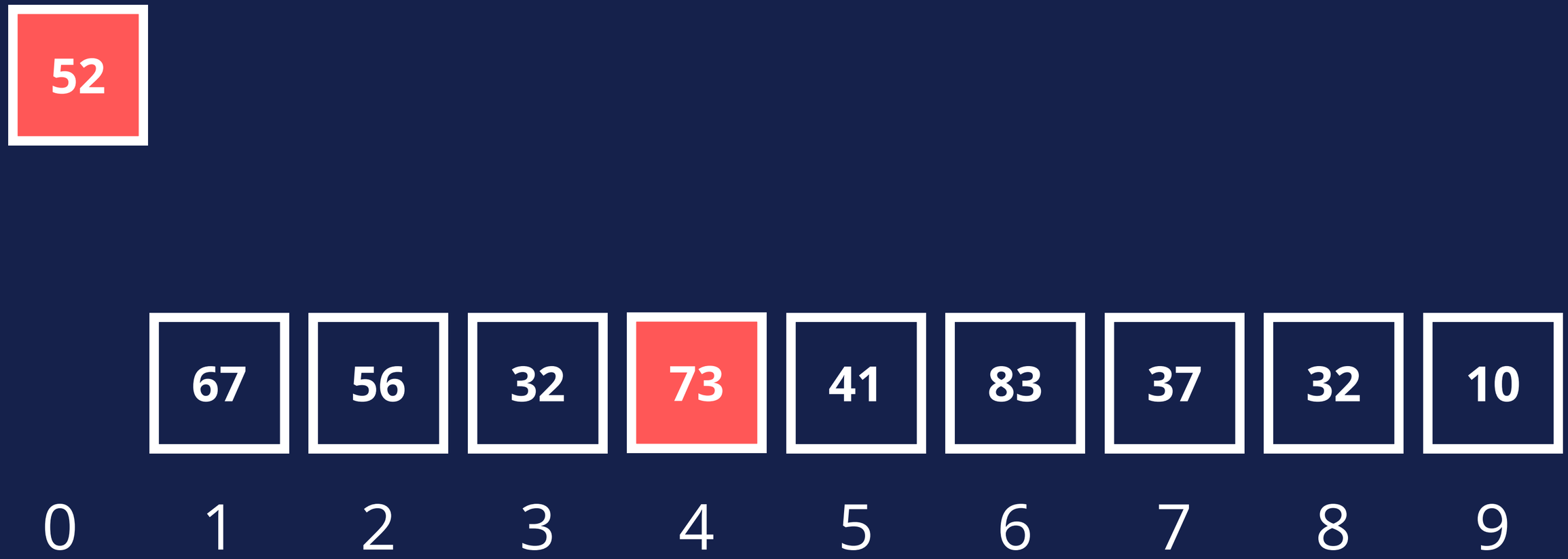
swaps = 0



Shell Sort using Knuth

interval = 4

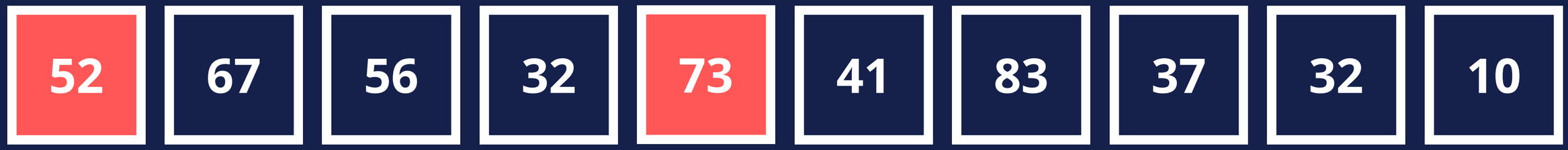
swaps = 1



Shell Sort using Knuth

interval = 4

swaps = 1

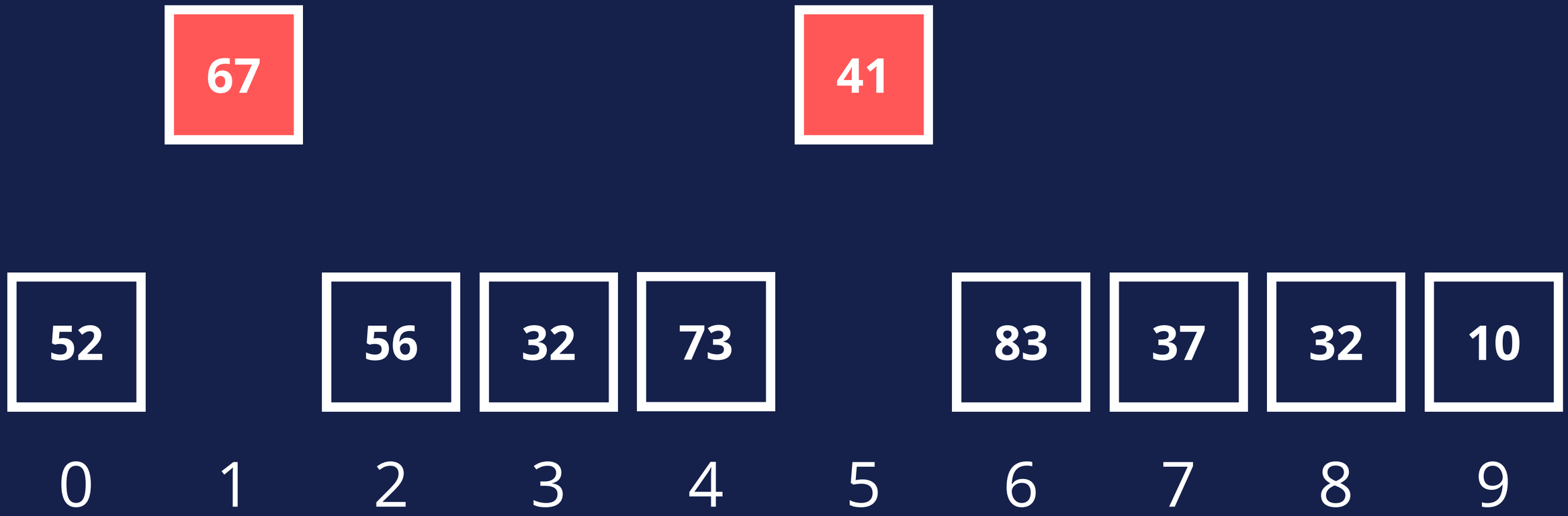


0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 4

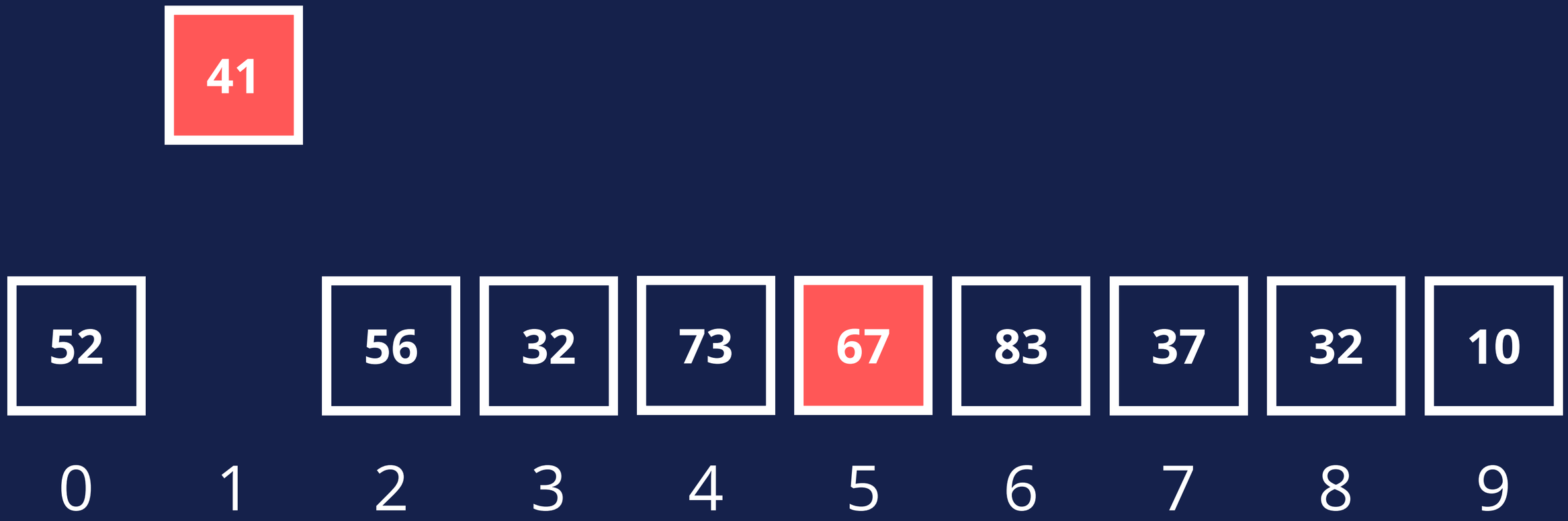
swaps = 1



Shell Sort using Knuth

interval = 4

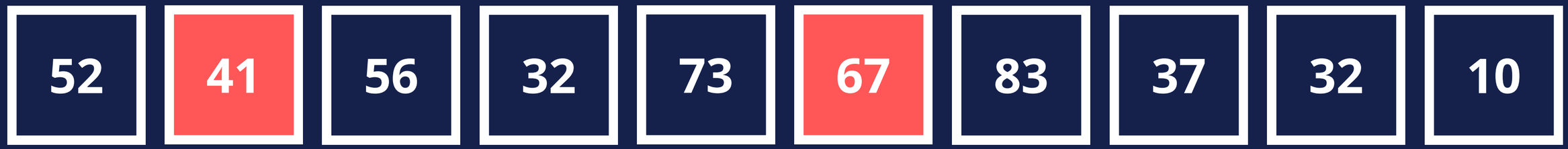
swaps = 2



Shell Sort using Knuth

interval = 4

swaps = 2

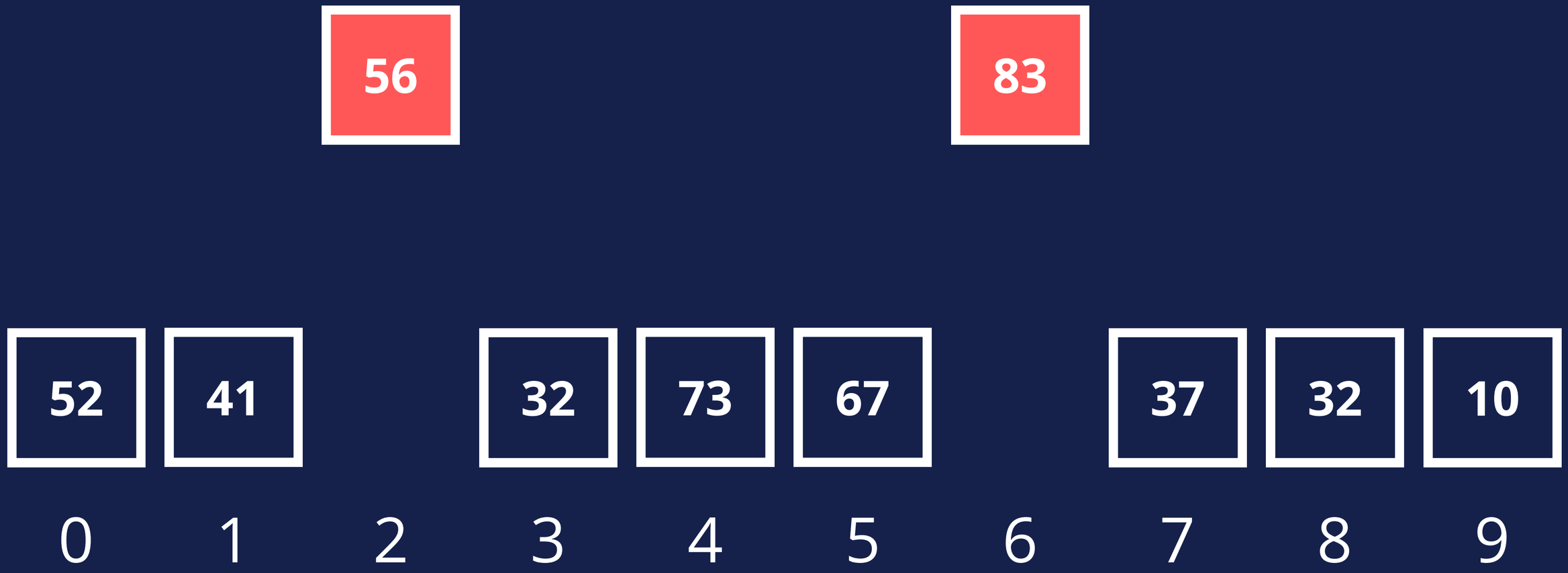


0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 4

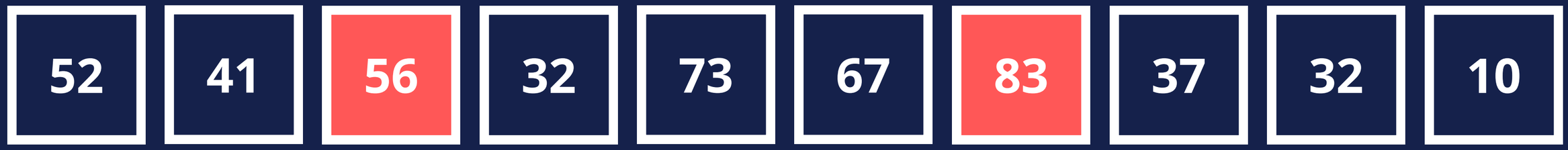
swaps = 2



Shell Sort using Knuth

interval = 4

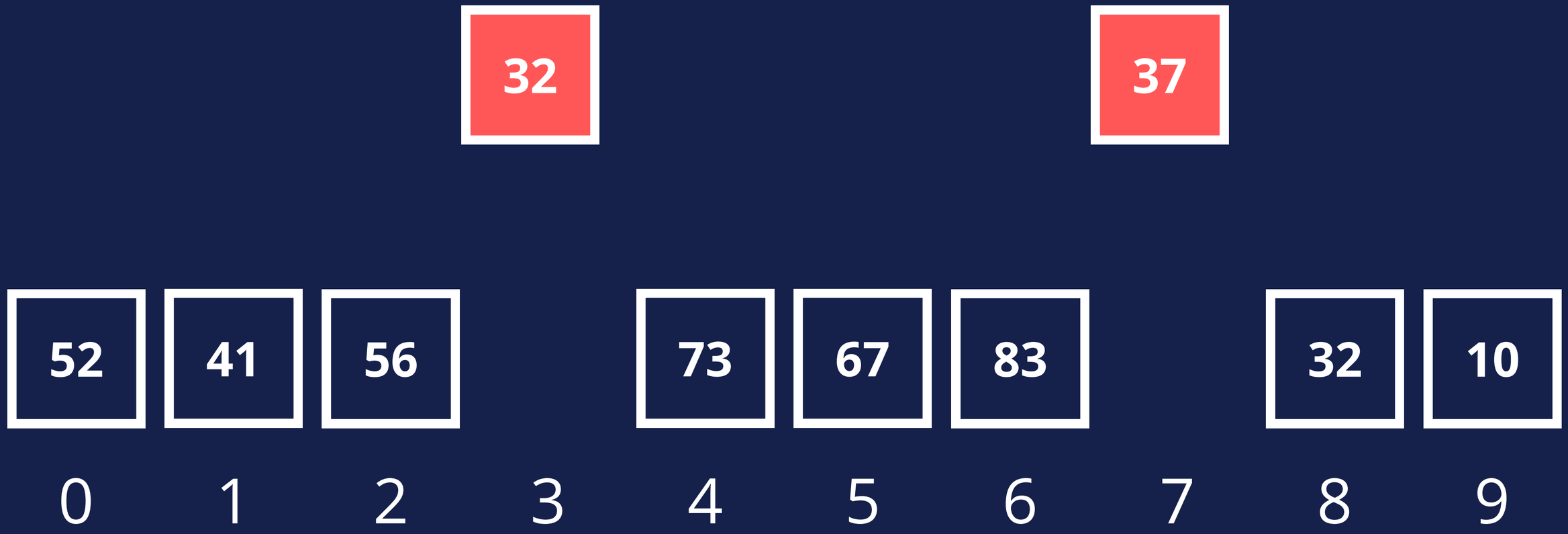
swaps = 2



Shell Sort using Knuth

interval = 4

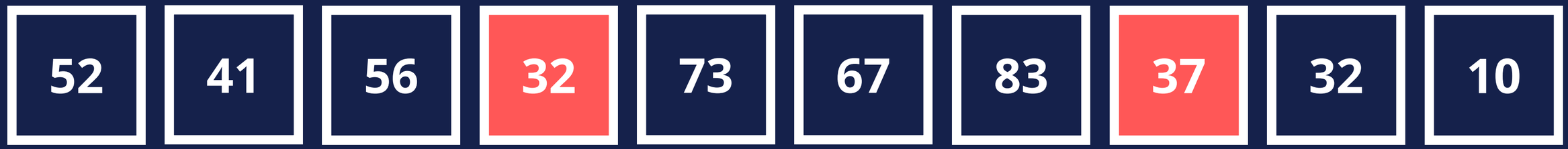
swaps = 2



Shell Sort using Knuth

interval = 4

swaps = 2

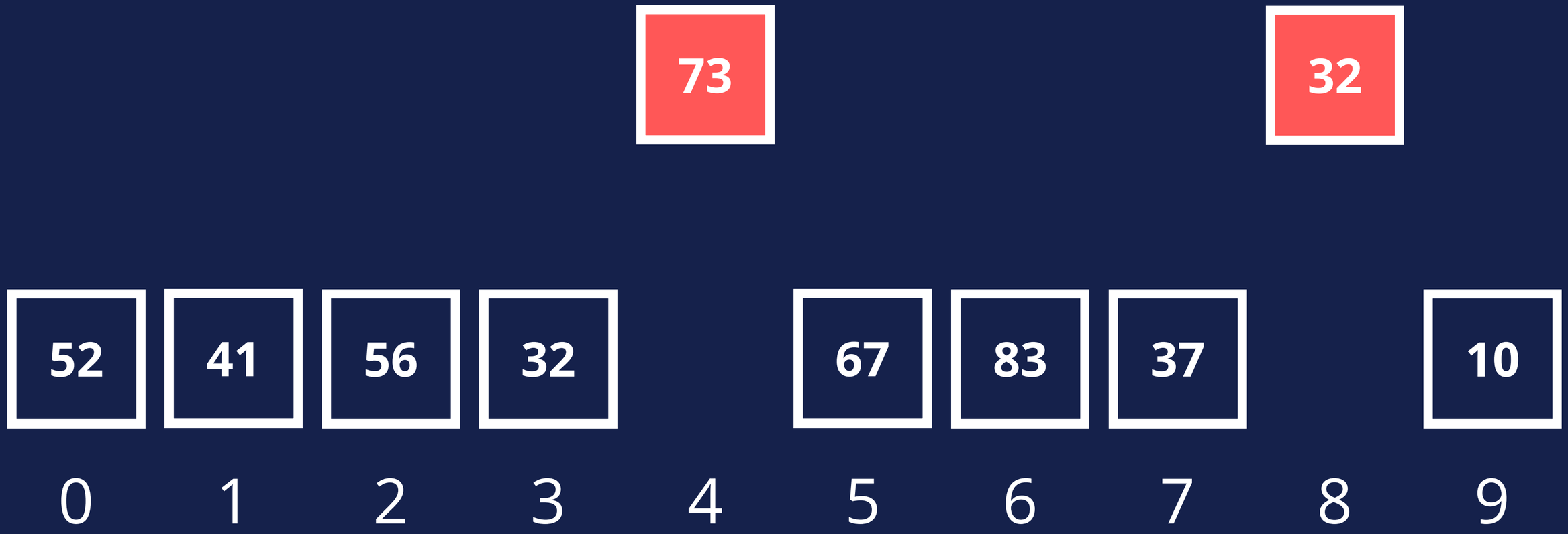


0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 4

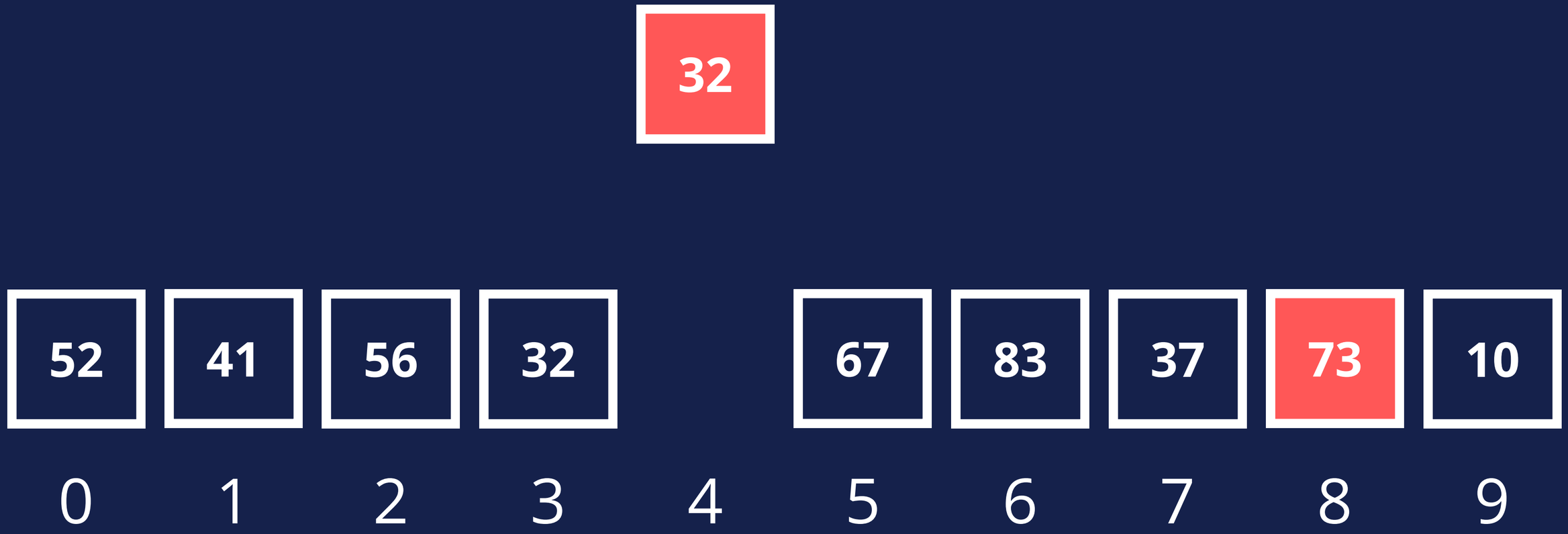
swaps = 2



Shell Sort using Knuth

interval = 4

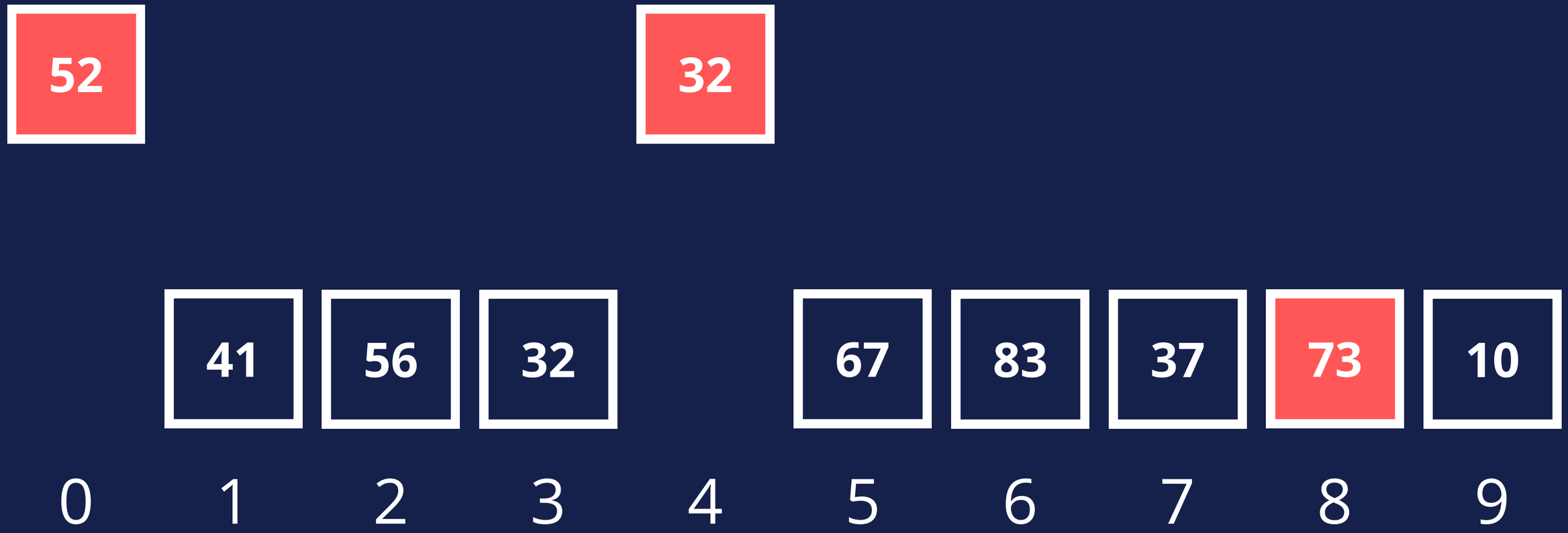
swaps = 3



Shell Sort using Knuth

interval = 4

swaps = 3



Shell Sort using Knuth

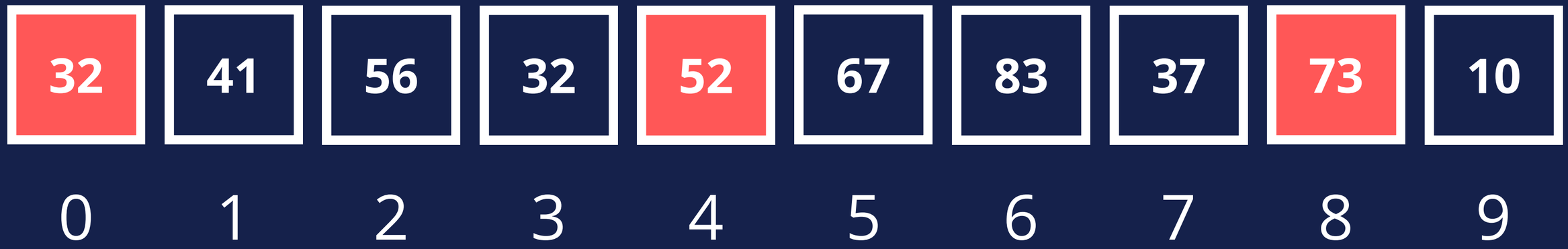
interval = 4

swaps = 4

Shell Sort using Knuth

interval = 4

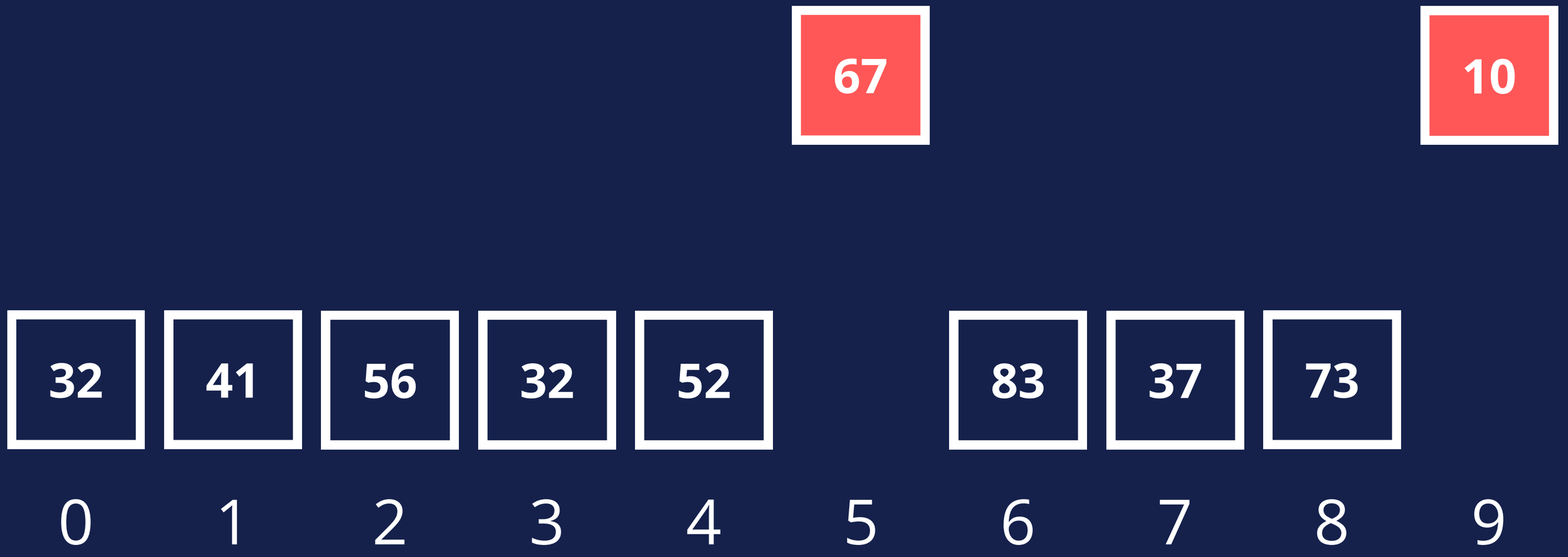
swaps = 4



Shell Sort using Knuth

interval = 4

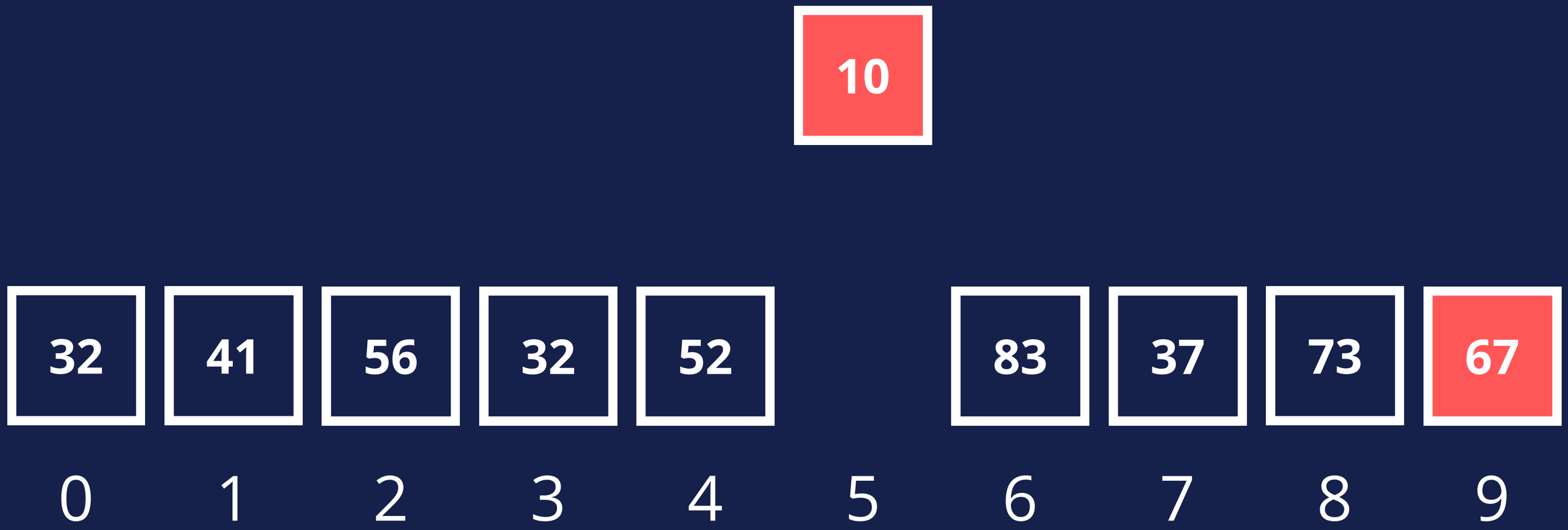
swaps = 4



Shell Sort using Knuth

interval = 4

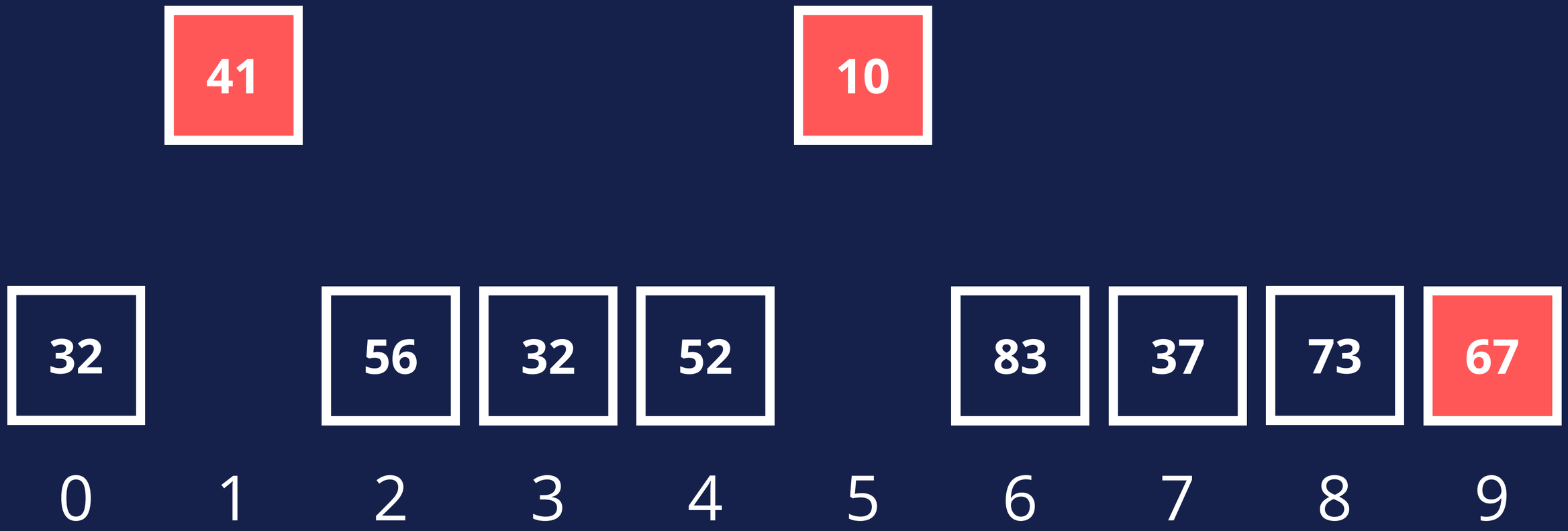
swaps = 5



Shell Sort using Knuth

interval = 4

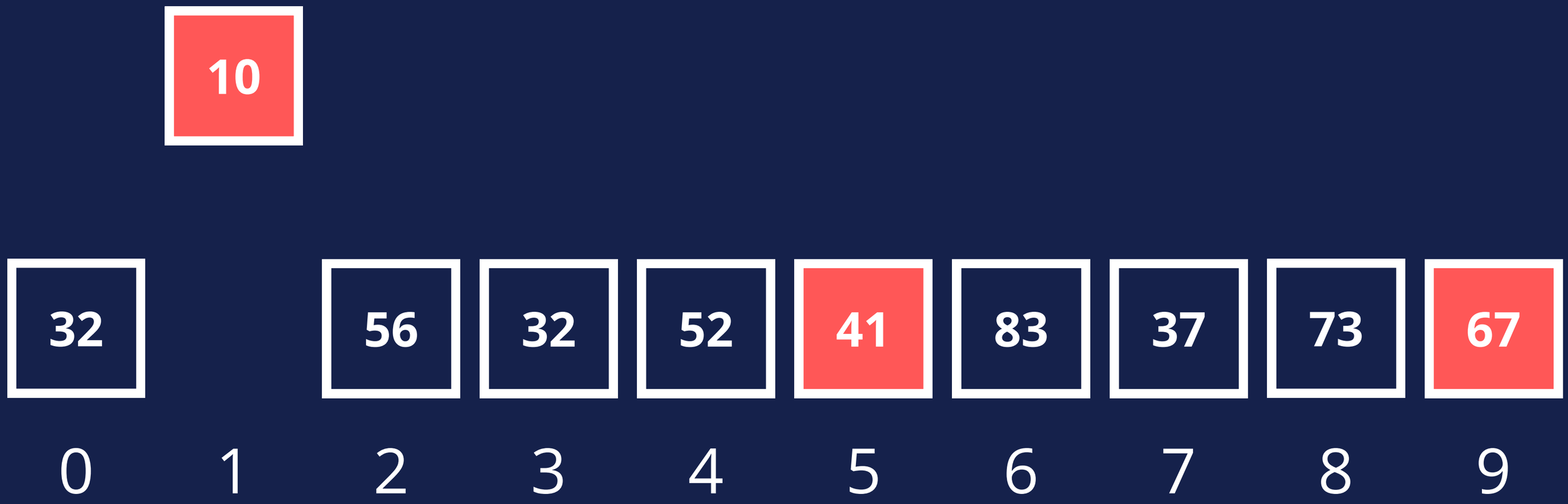
swaps = 5



Shell Sort using Knuth

interval = 4

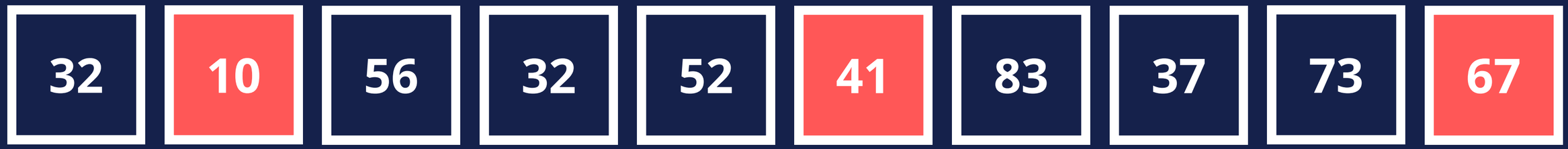
swaps = 6



Shell Sort using Knuth

interval = 4

swaps = 6



0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 4

swaps = 6

interval = interval / 3

0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 1

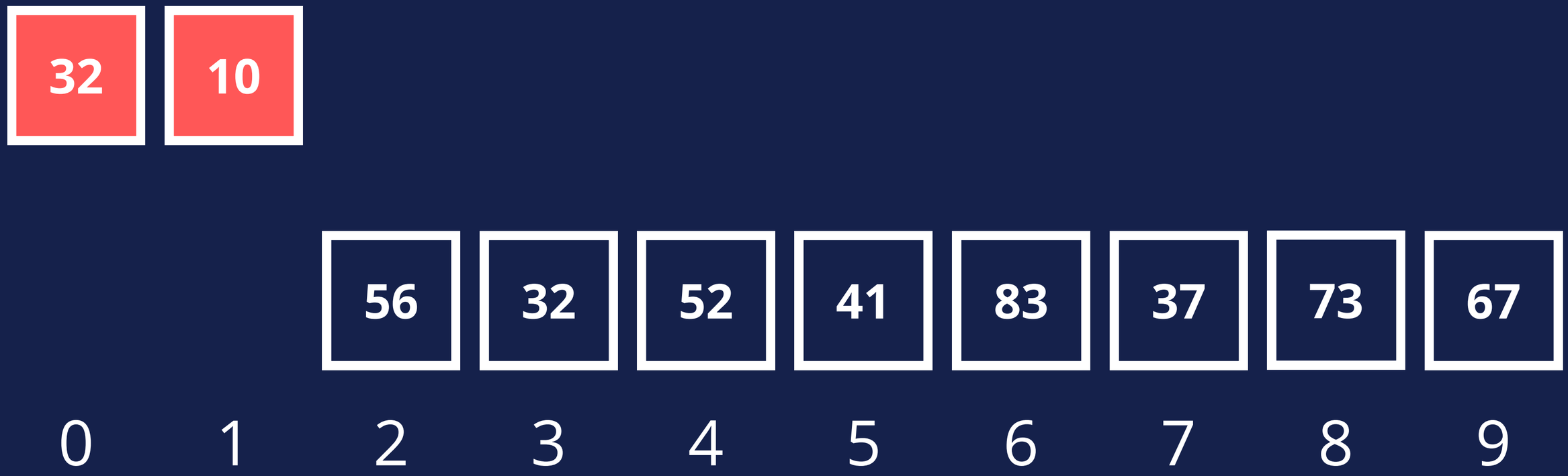
swaps = 6

0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 1

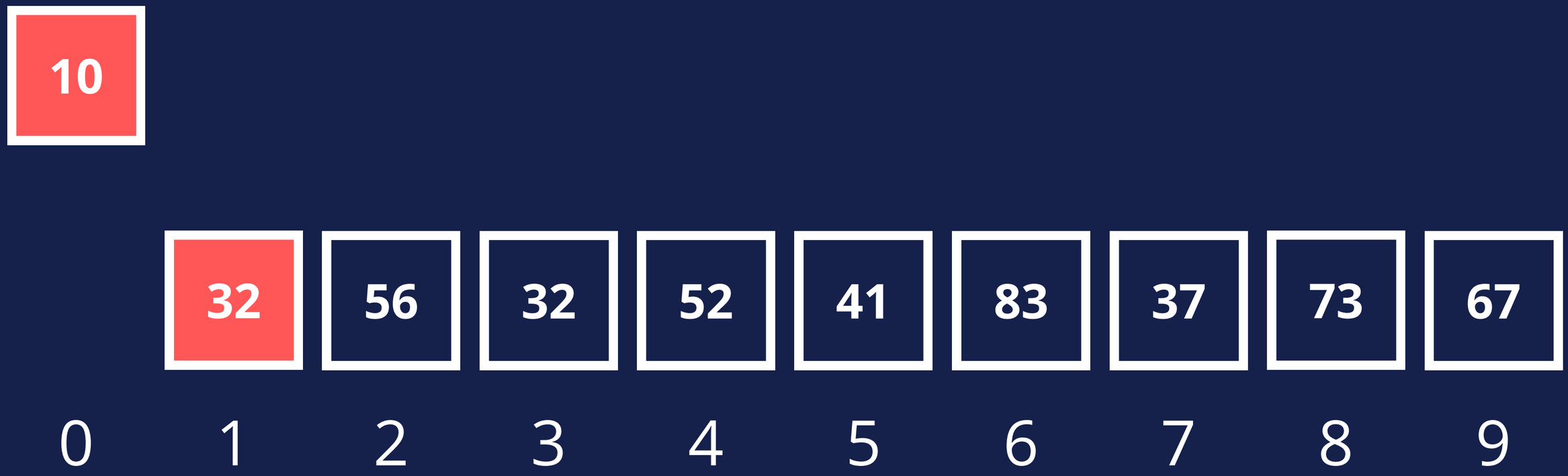
swaps = 6



Shell Sort using Knuth

interval = 1

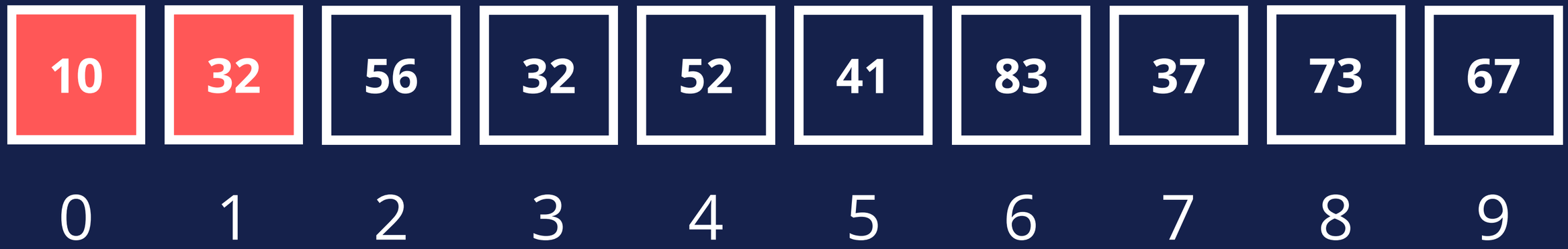
swaps = 7



Shell Sort using Knuth

interval = 1

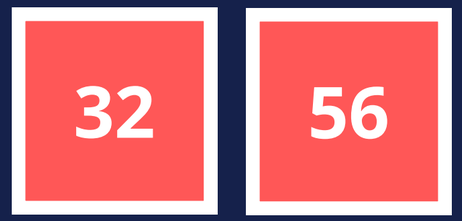
swaps = 7



Shell Sort using Knuth

interval = 1

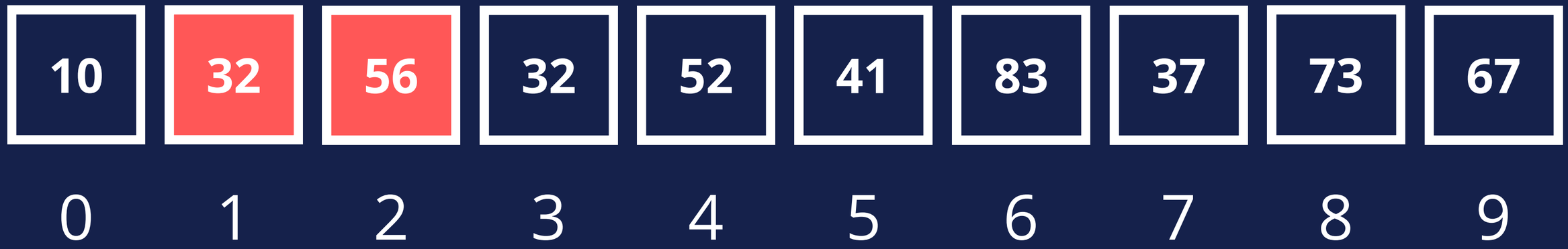
swaps = 7



Shell Sort using Knuth

interval = 1

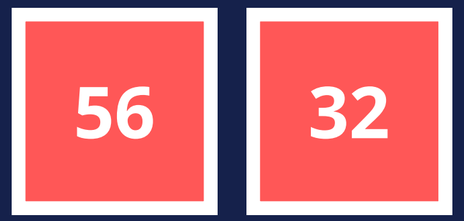
swaps = 7



Shell Sort using Knuth

interval = 1

swaps = 7

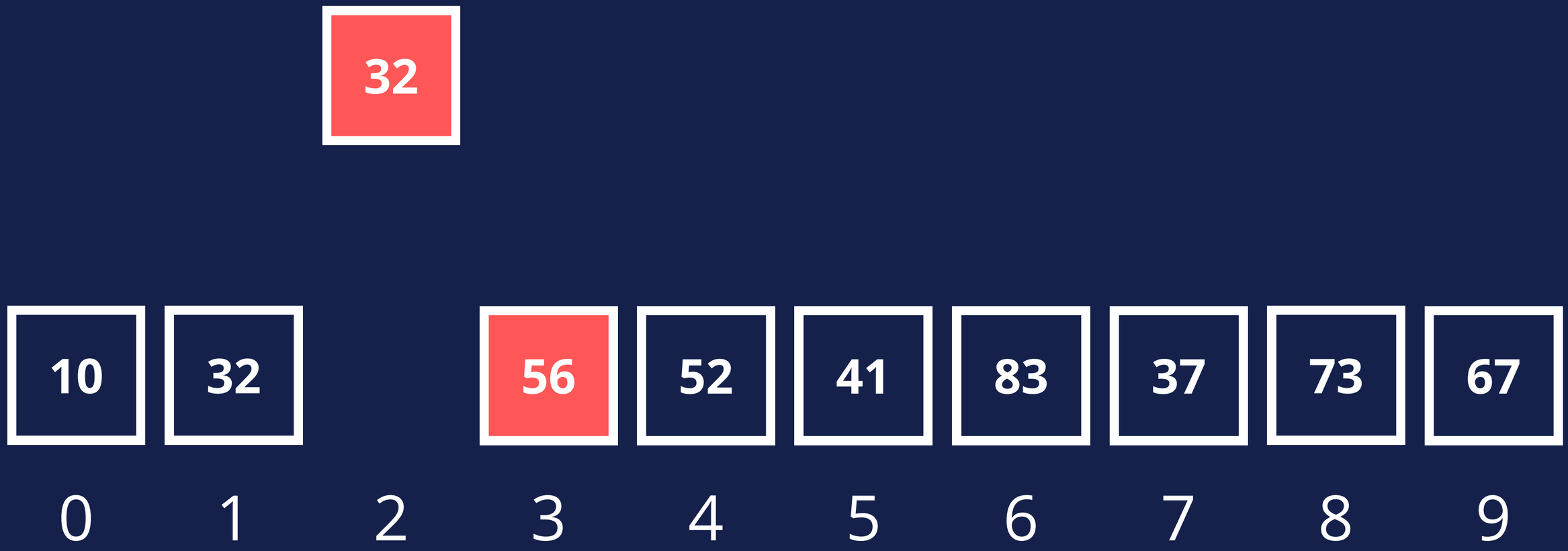


0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 1

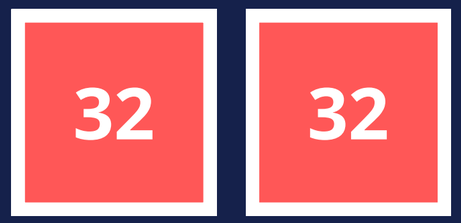
swaps = 8



Shell Sort using Knuth

interval = 1

swaps = 8



Shell Sort using Knuth

interval = 1

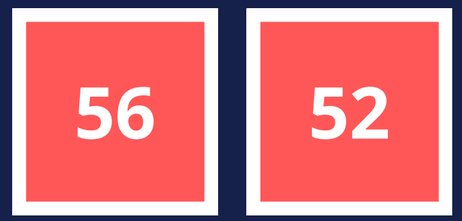
swaps = 8

0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 1

swaps = 8

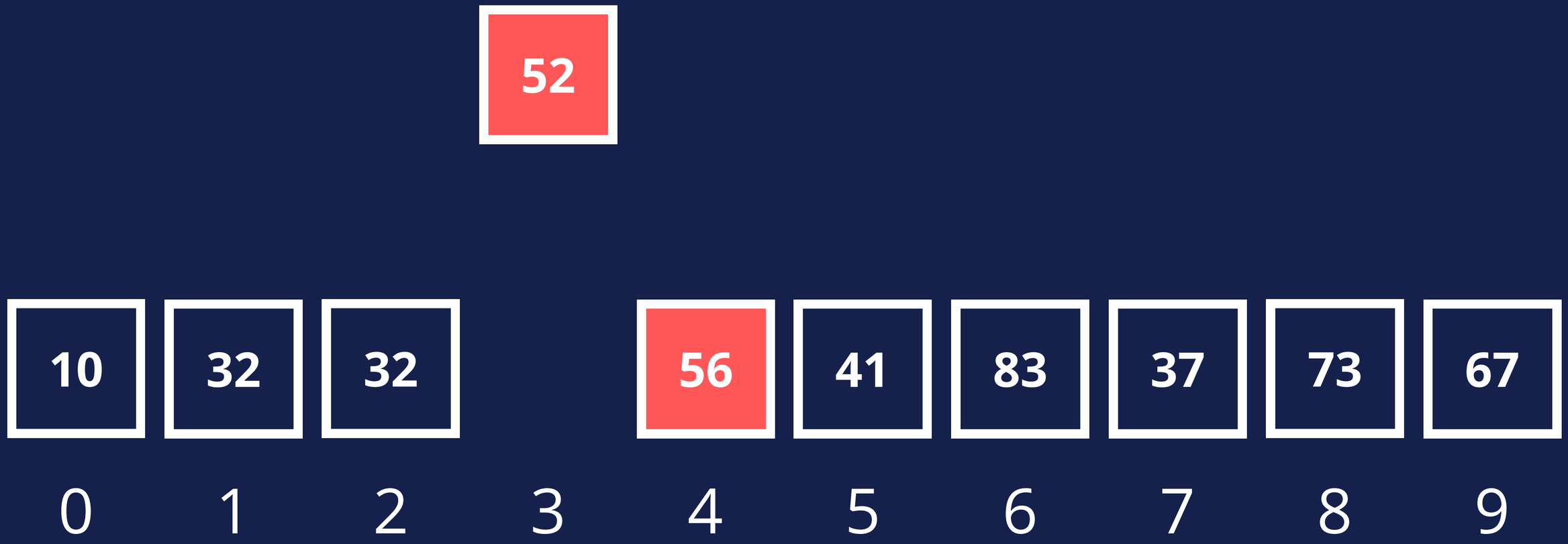


0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 1

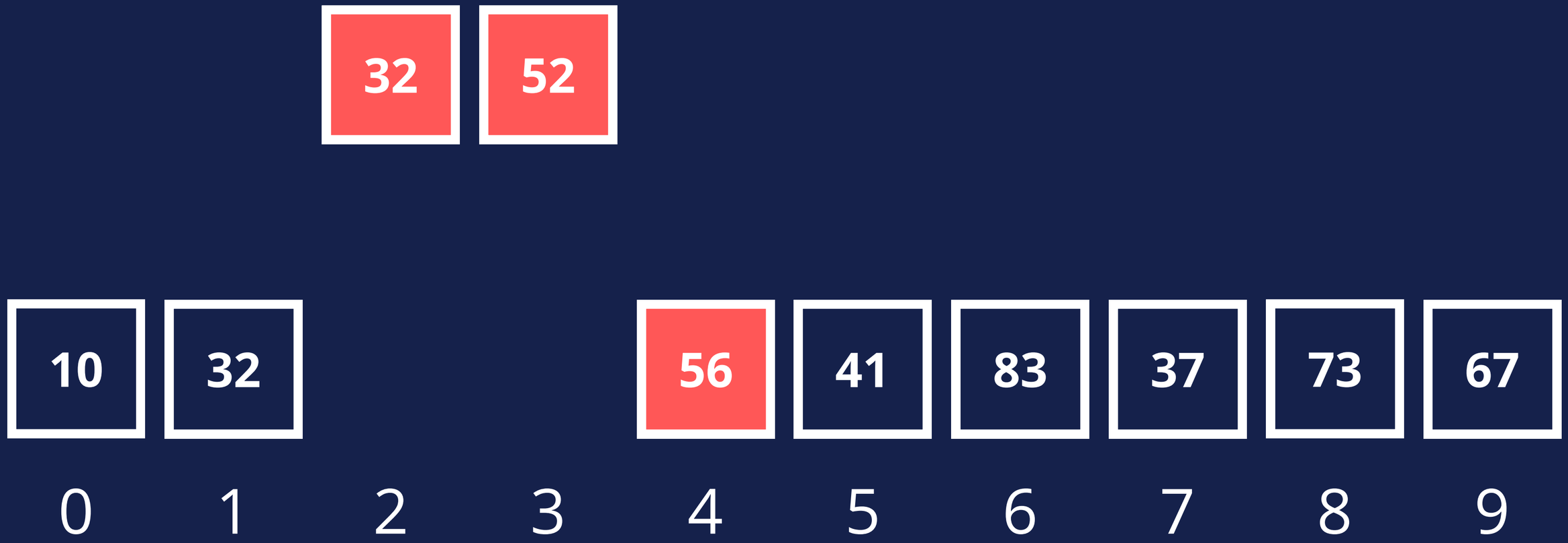
swaps = 9



Shell Sort using Knuth

interval = 1

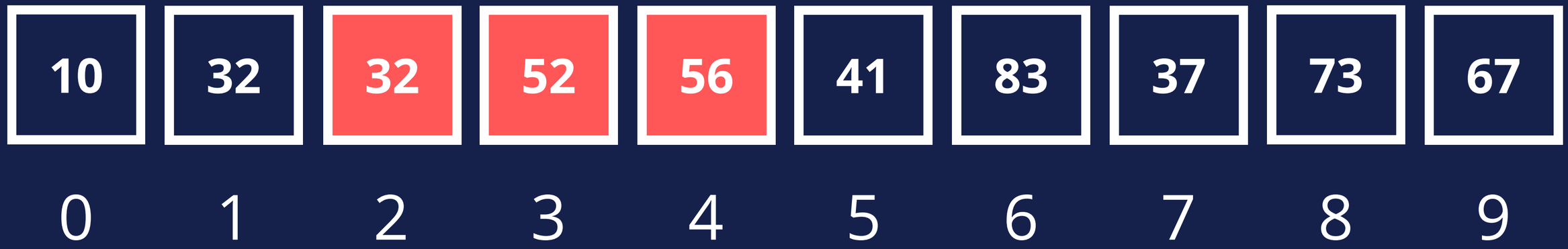
swaps = 9



Shell Sort using Knuth

interval = 1

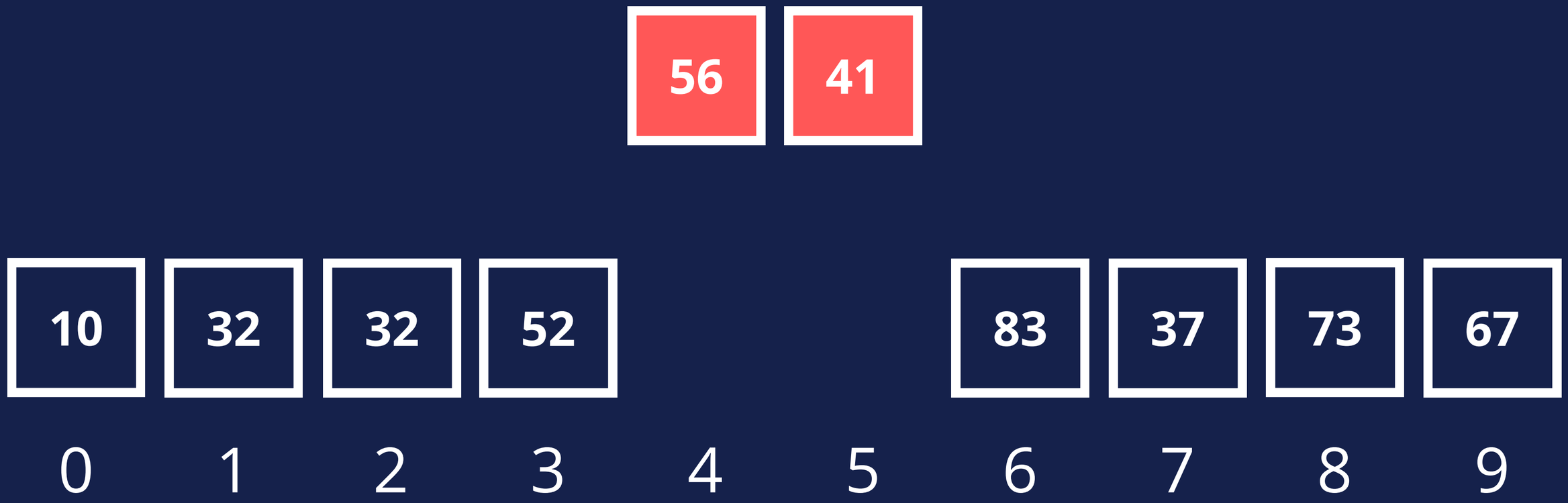
swaps = 9



Shell Sort using Knuth

interval = 1

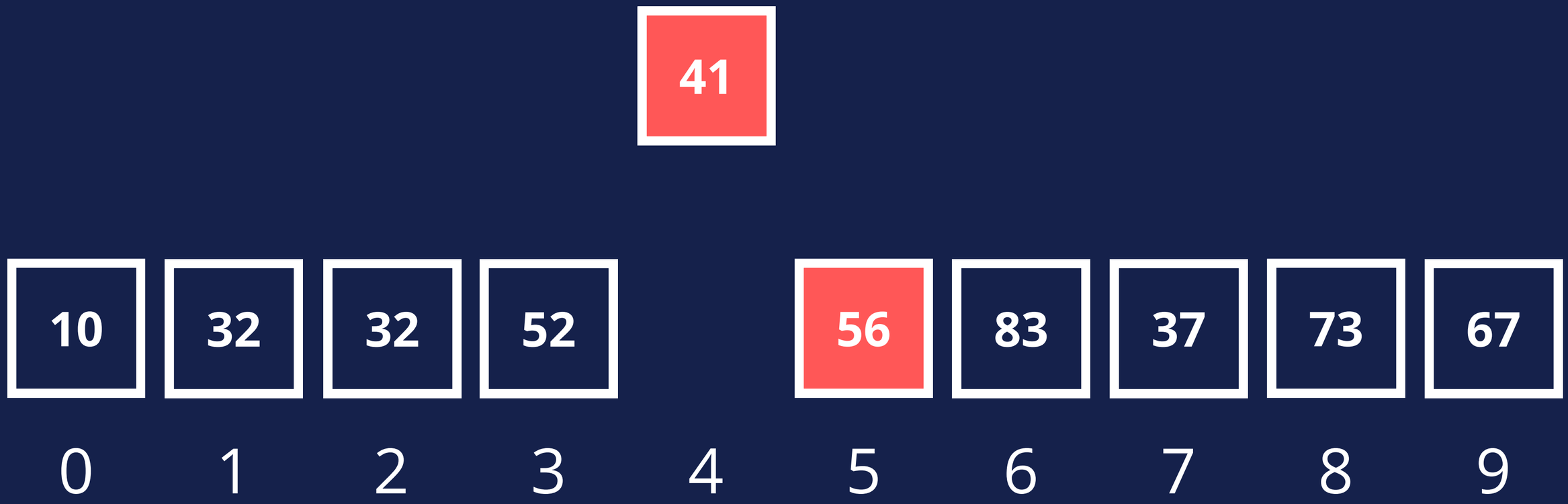
swaps = 9



Shell Sort using Knuth

interval = 1

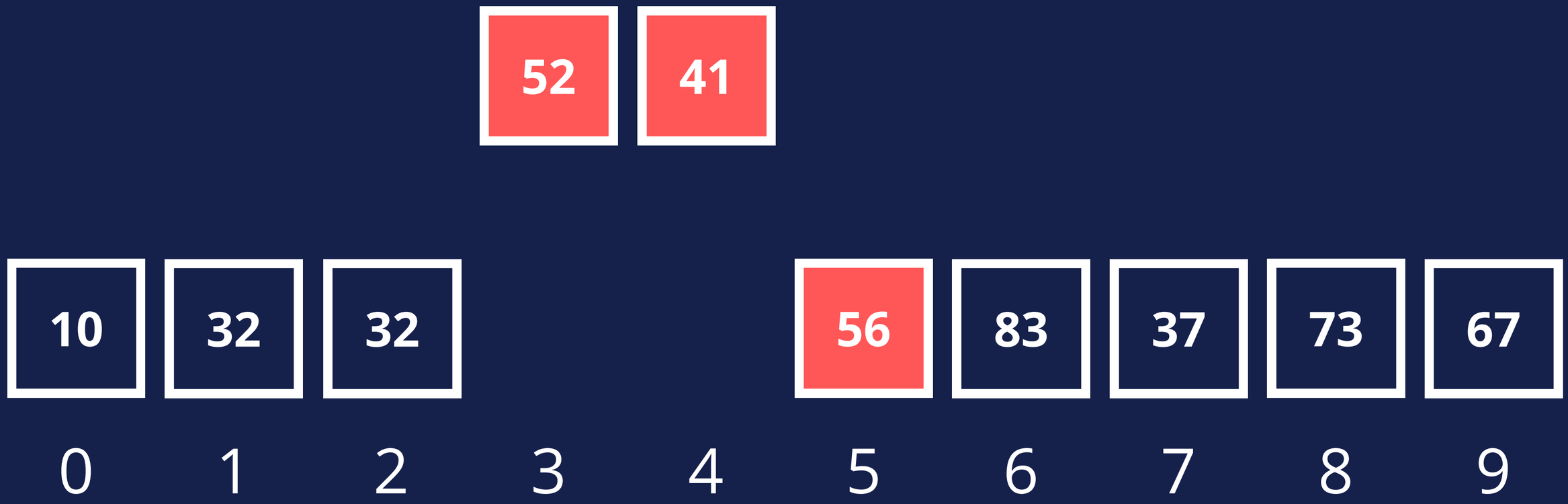
swaps = 10



Shell Sort using Knuth

interval = 1

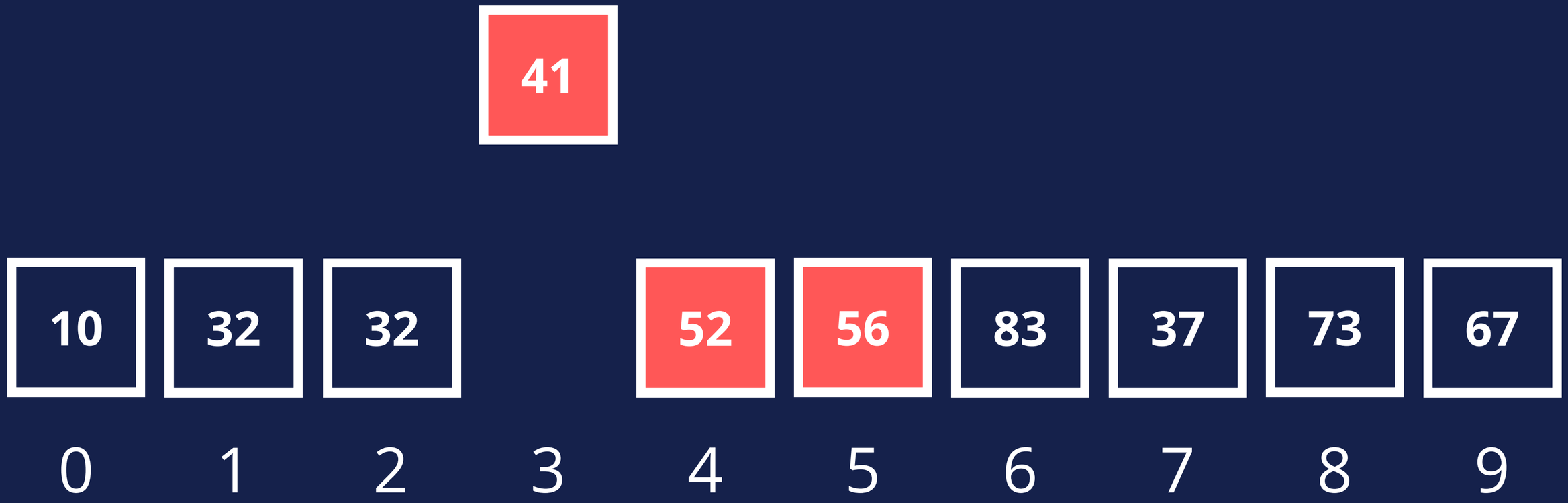
swaps = 10



Shell Sort using Knuth

interval = 1

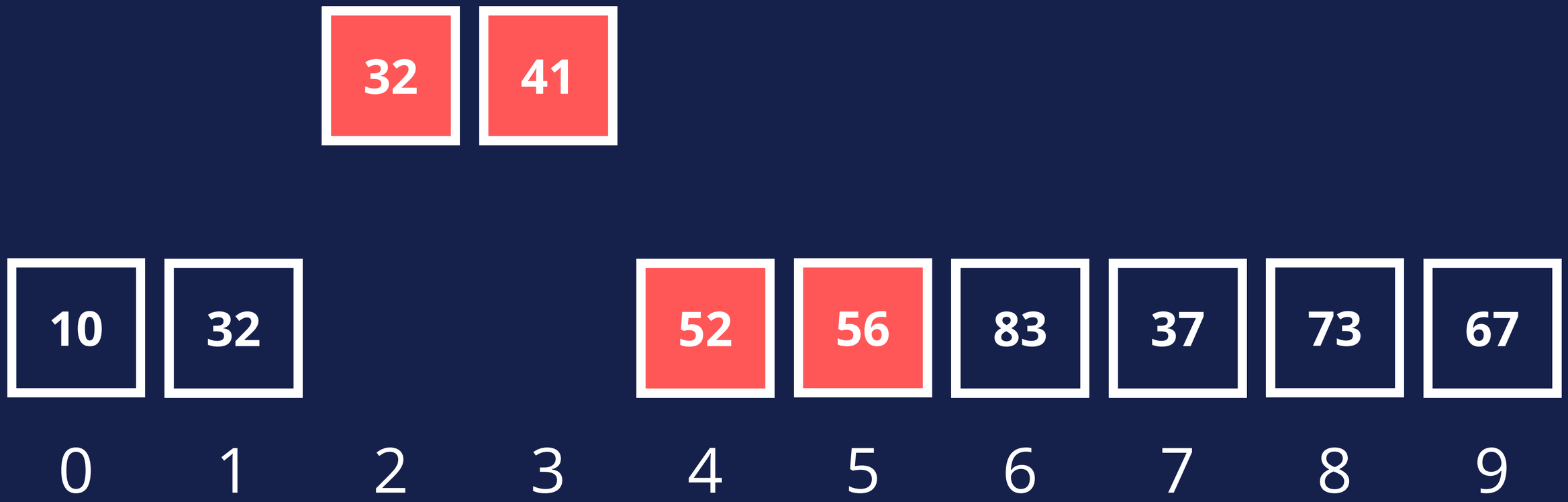
swaps = 11



Shell Sort using Knuth

interval = 1

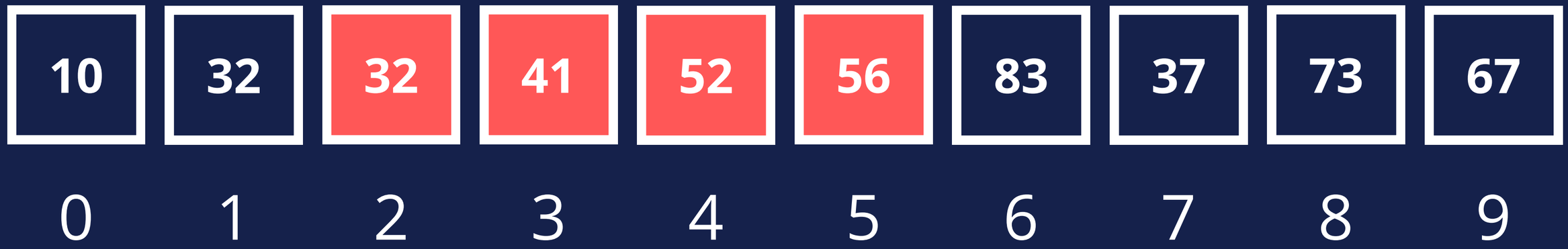
swaps = 11



Shell Sort using Knuth

interval = 1

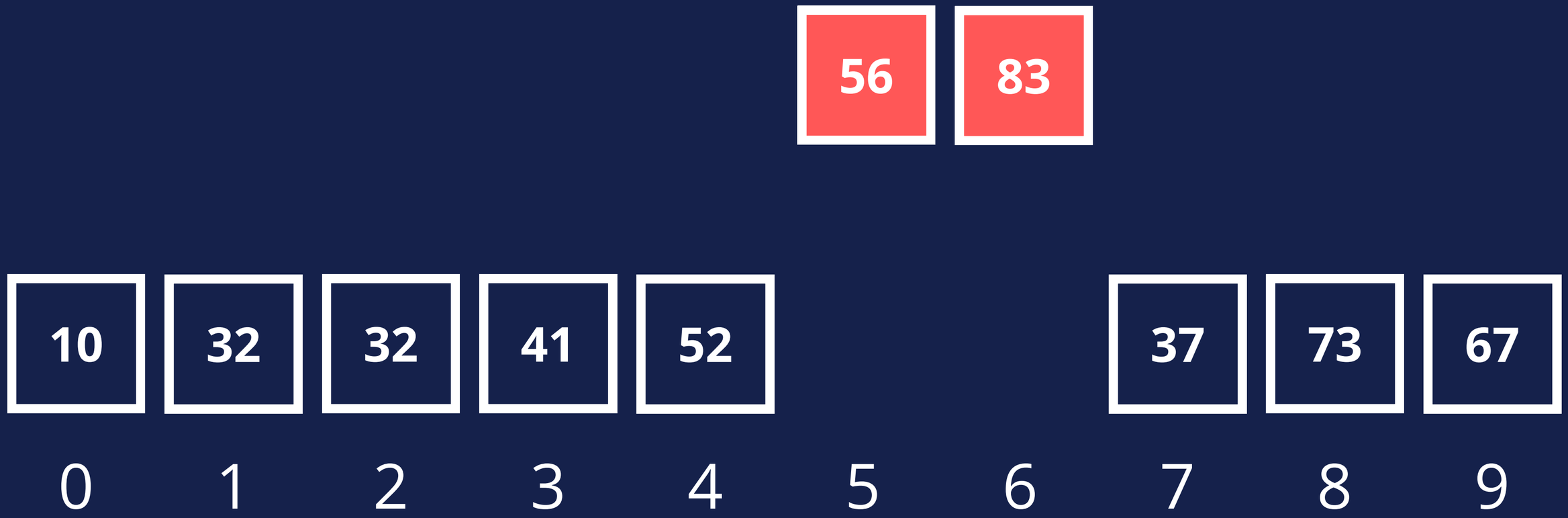
swaps = 11



Shell Sort using Knuth

interval = 1

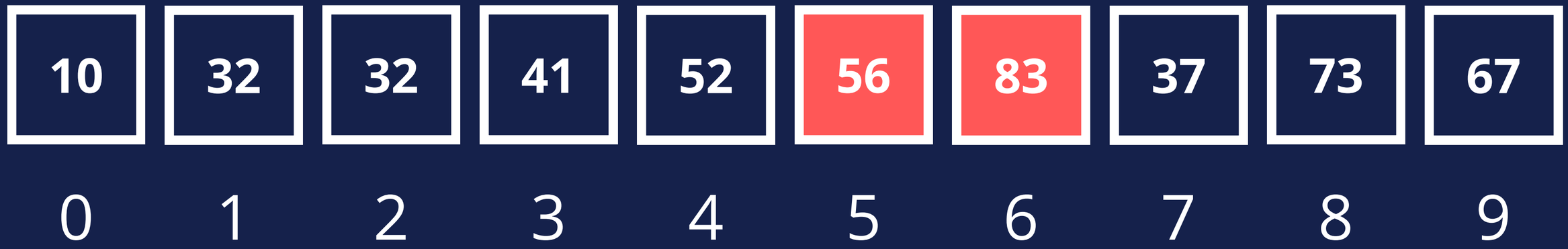
swaps = 11



Shell Sort using Knuth

interval = 1

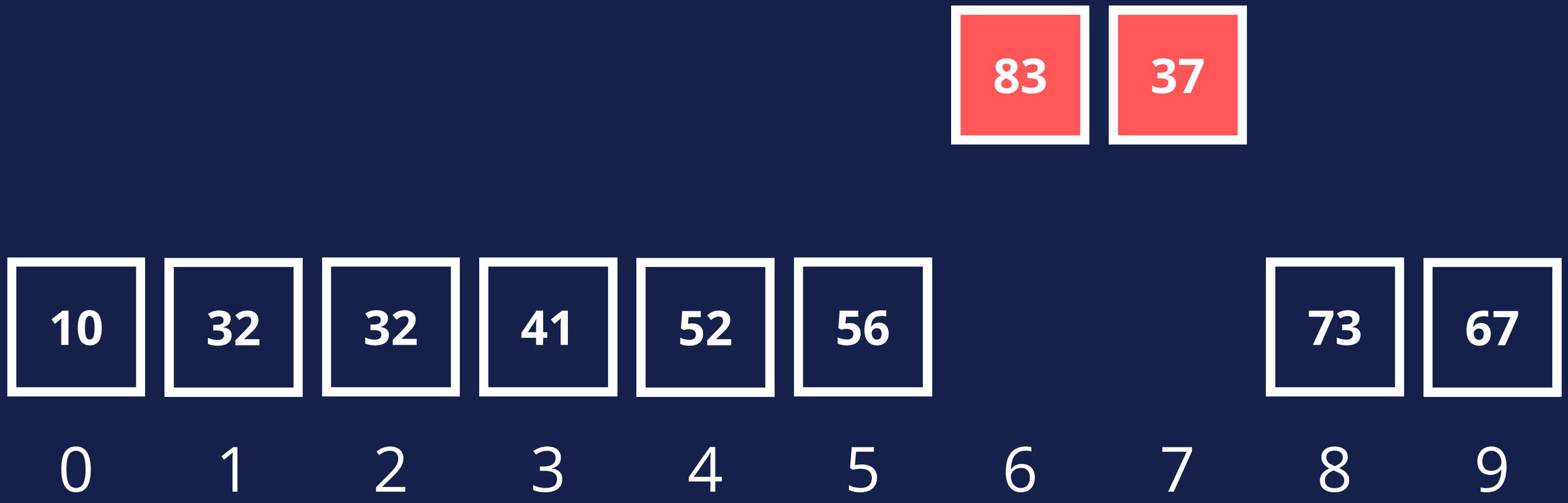
swaps = 11



Shell Sort using Knuth

interval = 1

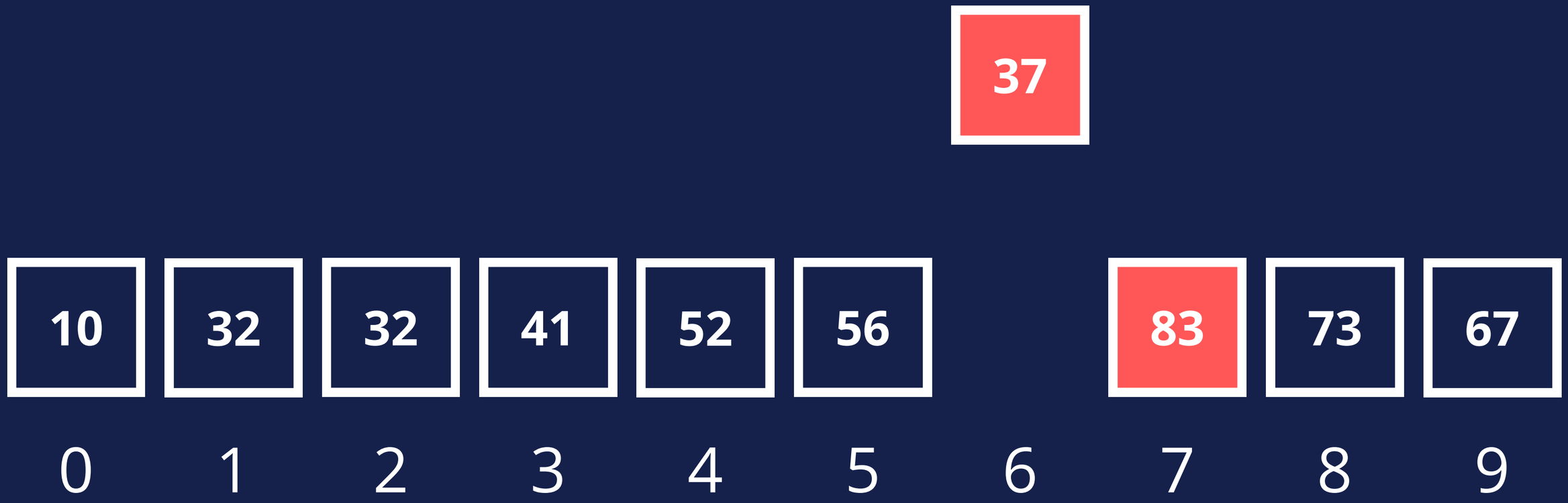
swaps = 11



Shell Sort using Knuth

interval = 1

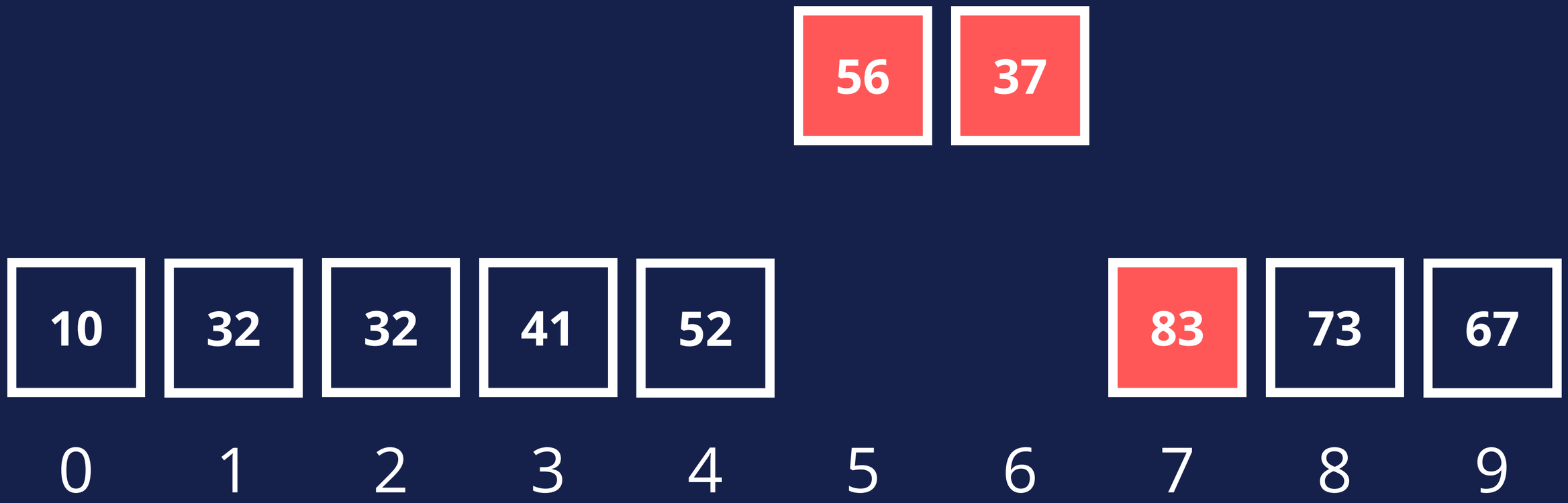
swaps = 12



Shell Sort using Knuth

interval = 1

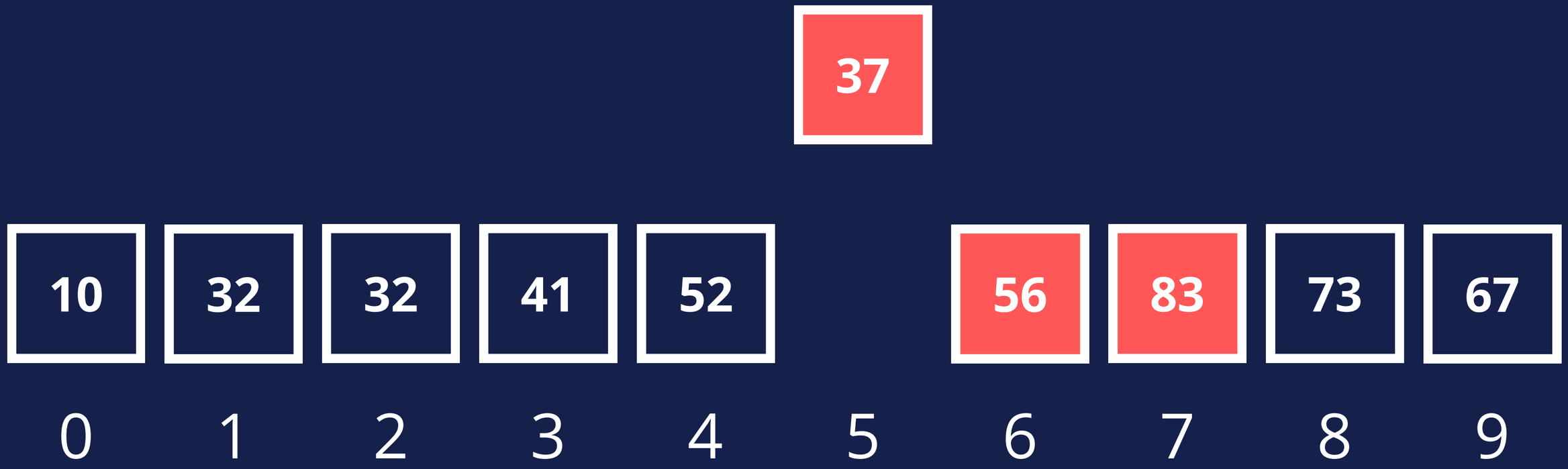
swaps = 12



Shell Sort using Knuth

interval = 1

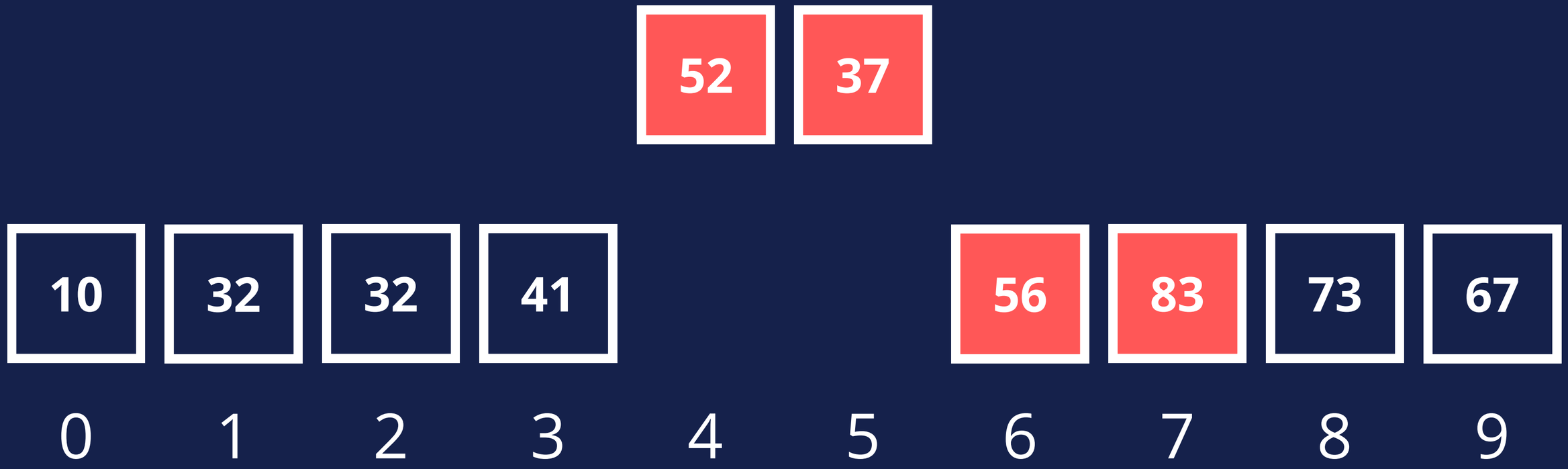
swaps = 13



Shell Sort using Knuth

interval = 1

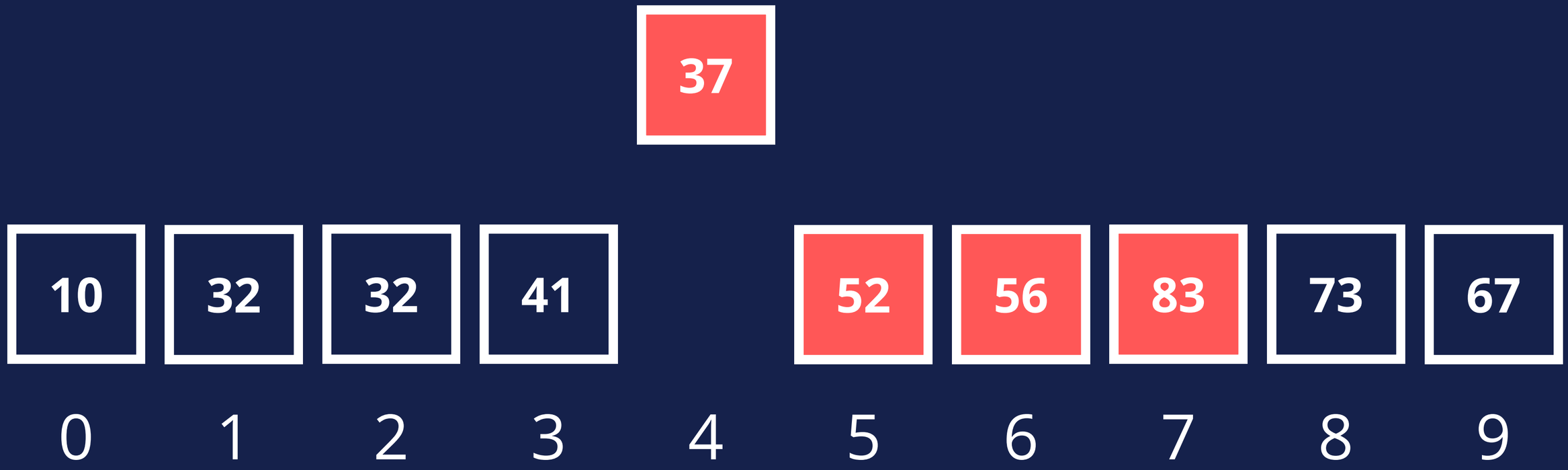
swaps = 13



Shell Sort using Knuth

interval = 1

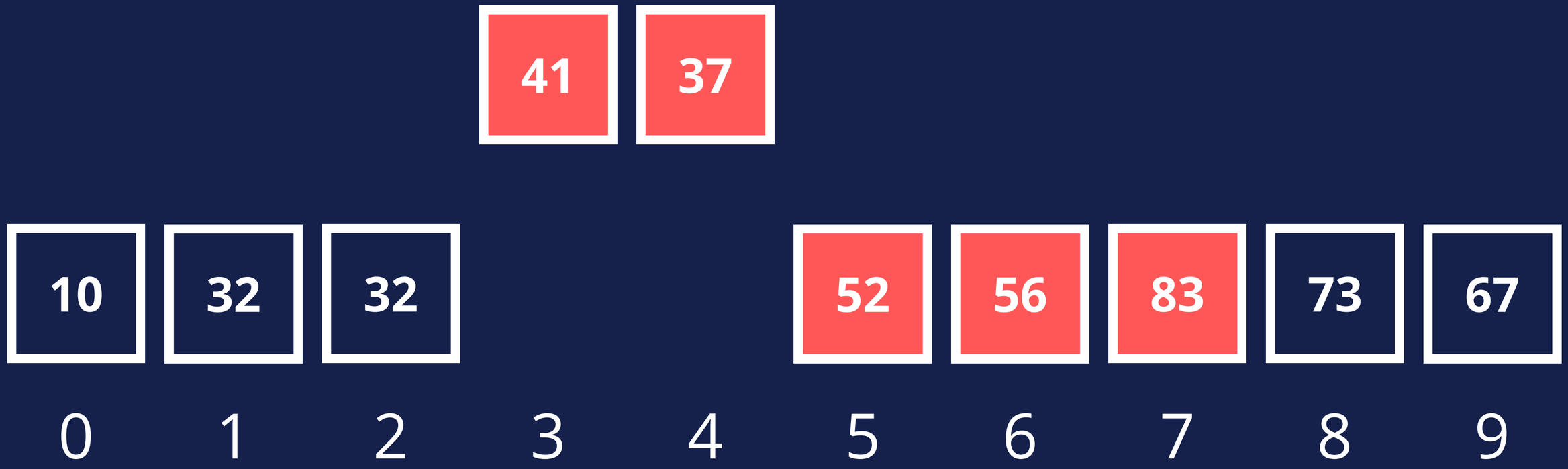
swaps = 14



Shell Sort using Knuth

interval = 1

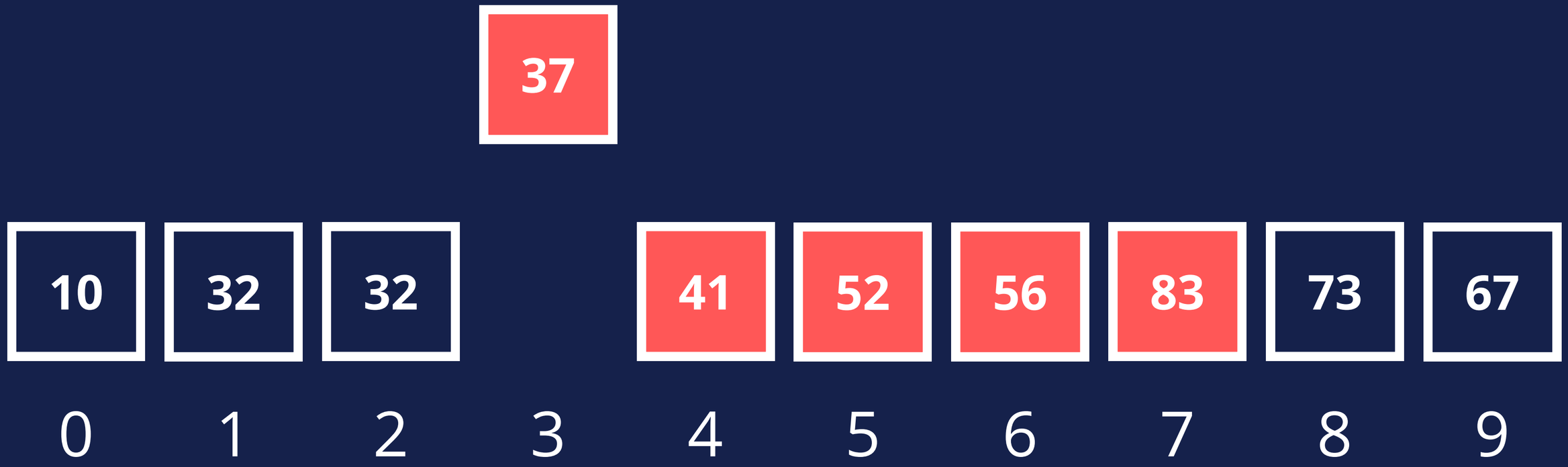
swaps = 14



Shell Sort using Knuth

interval = 1

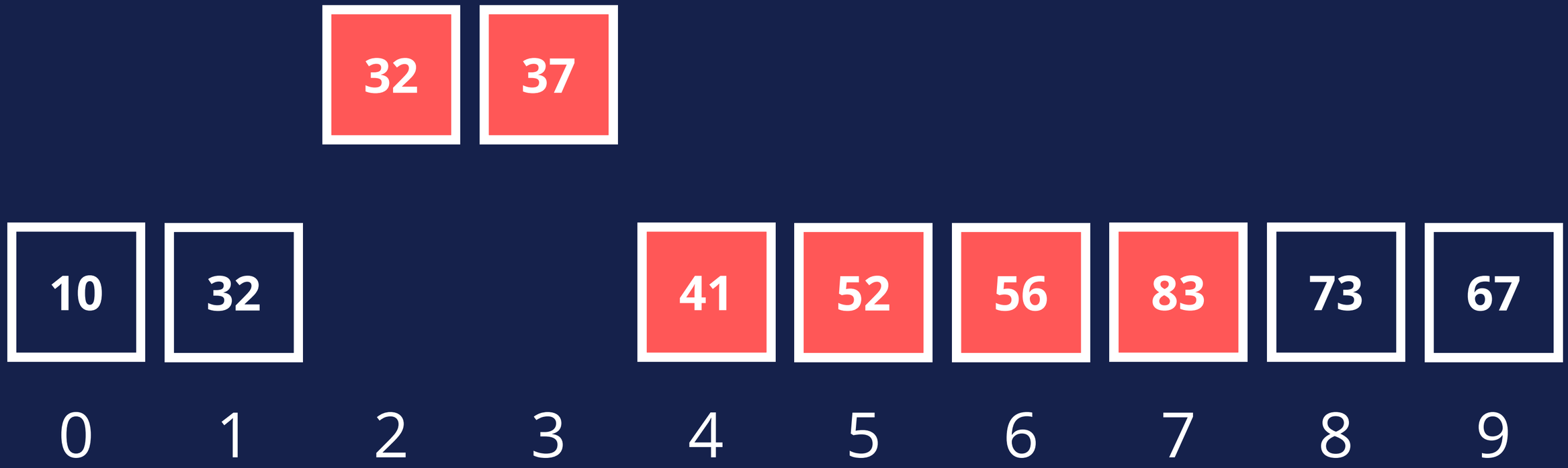
swaps = 15



Shell Sort using Knuth

interval = 1

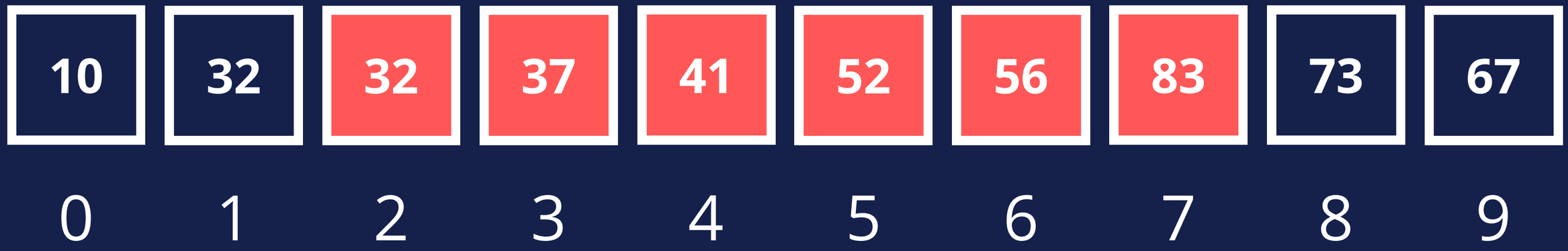
swaps = 15



Shell Sort using Knuth

interval = 1

swaps = 15



Shell Sort using Knuth

interval = 1

swaps = 15

Shell Sort using Knuth

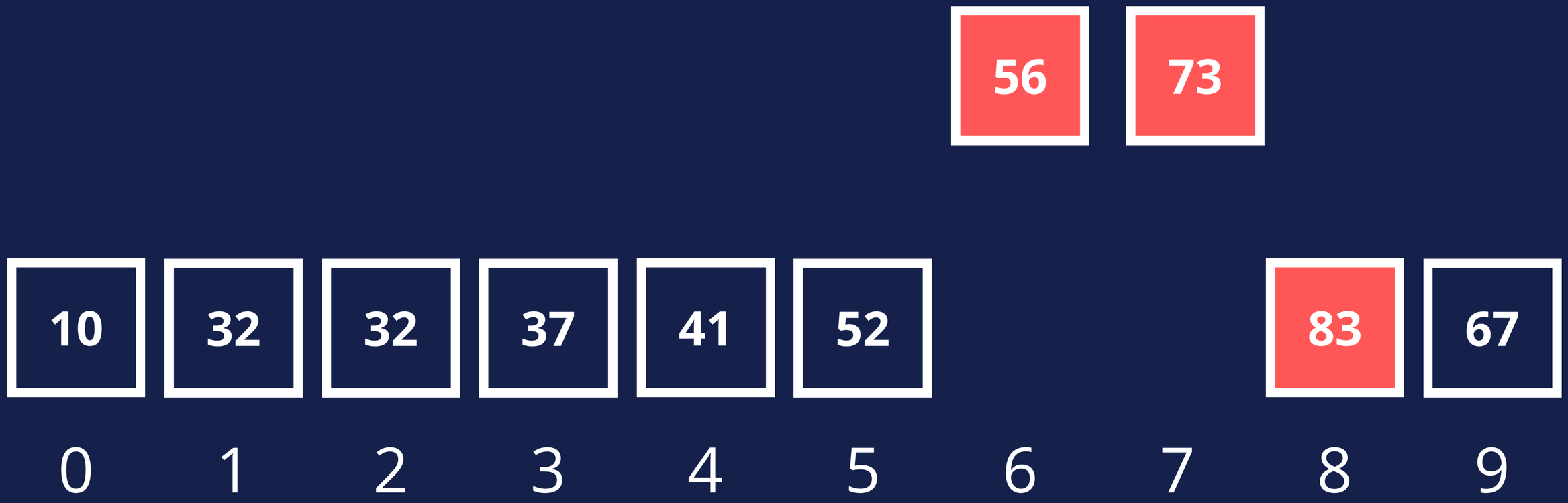
interval = 1

swaps = 16

Shell Sort using Knuth

interval = 1

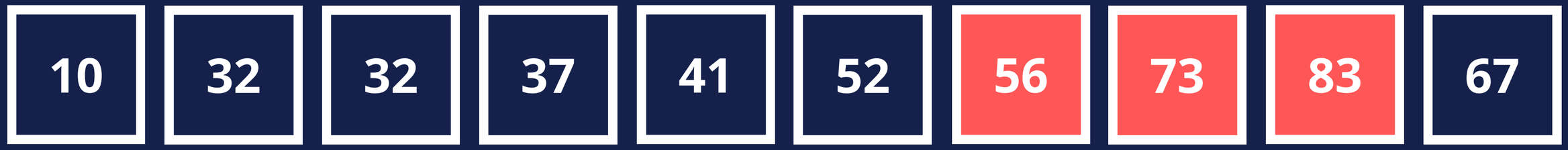
swaps = 16



Shell Sort using Knuth

interval = 1

swaps = 16

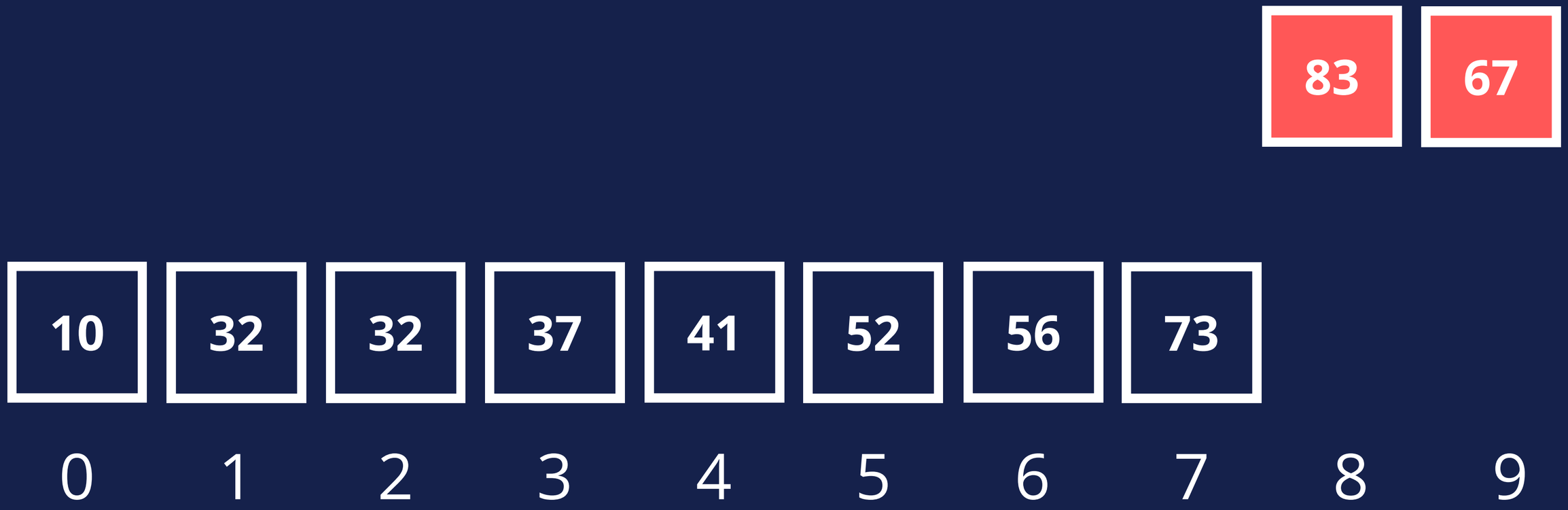


0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 1

swaps = 16



Shell Sort using Knuth

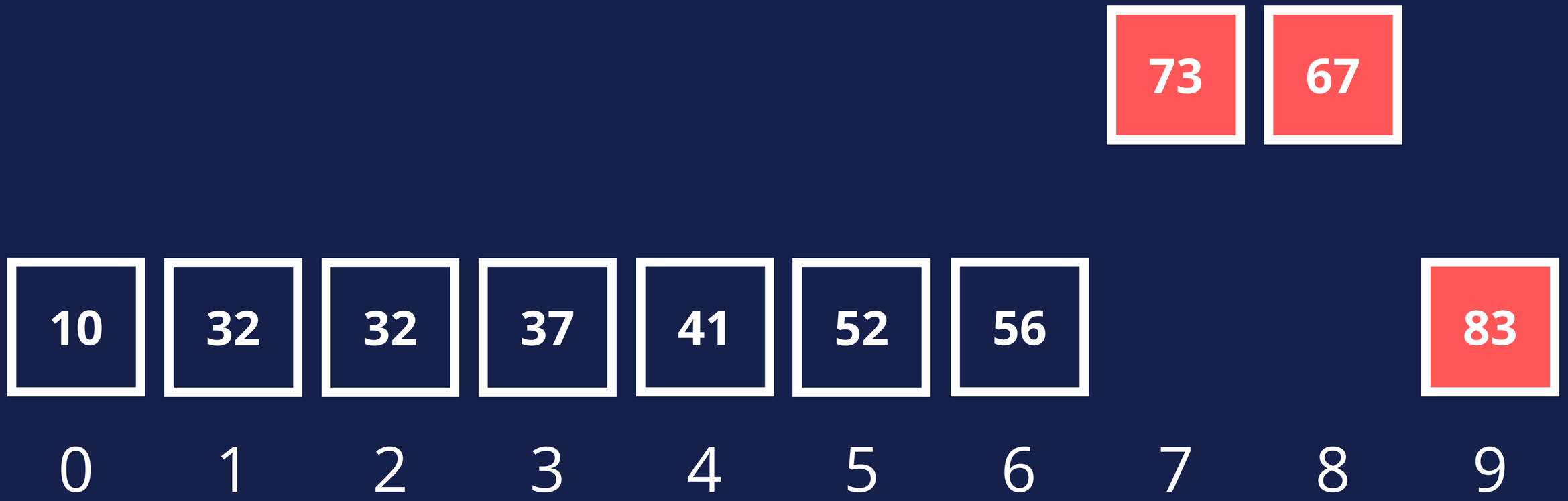
interval = 1

swaps = 17

Shell Sort using Knuth

interval = 1

swaps = 17



Shell Sort using Knuth

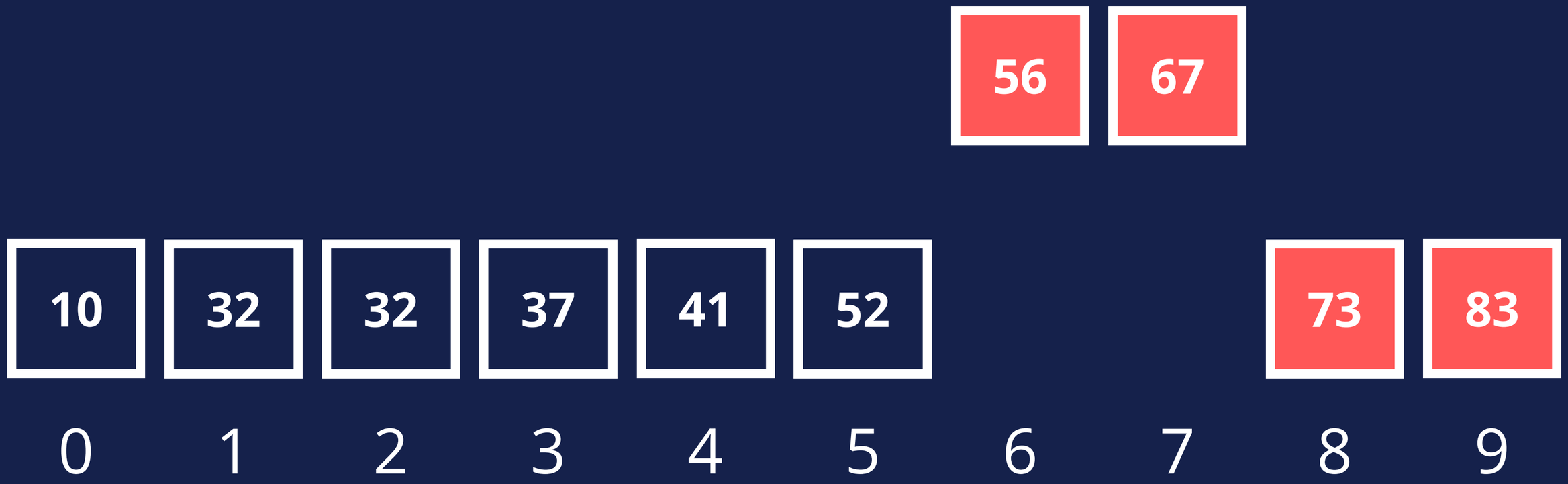
interval = 1

swaps = 18

Shell Sort using Knuth

interval = 1

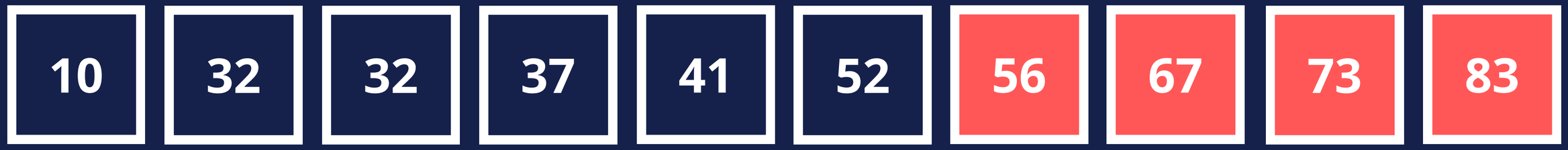
swaps = 18



Shell Sort using Knuth

interval = 1

swaps = 18



0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 1

swaps = 18

interval = interval / 3

0 1 2 3 4 5 6 7 8 9

Shell Sort using Knuth

interval = 0

swaps = 18

0 1 2 3 4 5 6 7 8 9