**Міністерство Освіти І НАУКИ України**

**Національний університет "Львівська політехніка"**

Інститут **ІКНІ**

Кафедра **ПЗ**

### ЗВІТ

До лабораторної роботи № 4

**З дисципліни:** *“Алгоритми та структури даних”*

**На тему:** *“Метод сортування злиттям”*

**Лектор:**

доц. каф. ПЗ

Коротєєва Т.О.

**Виконав:**

ст. гр. ПЗ-26

Матвіїв Т.Т.

**Прийняв:**

асист. каф. ПЗ

Франко А.В.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2020 р.

∑= \_\_\_\_\_ .

Львів – 2020

Тема роботи: Метод сортування злиттям.

Мета роботи: Вивчити алгоритм сортування злиттям. Здійснити програмну реалізацію алгоритму сортування злиттям. Дослідити швидкодію алгоритму.

TЕОРЕТИЧНІ ВІДОМОСТІ

Сортування злиттям (англійською «Merge Sort») — алгоритм сортування, в основі якого лежить принцип «розділяй та володарюй» (англійською «Divide and Conquer»). В основі цього способу сортування лежить злиття двох упорядкованих ділянок масиву в одну впорядковану ділянку іншого масиву.

Підчас сортування в дві допоміжні послідовності з основної поміщаються перші дві відсортовані підпослідовності, які потім зливаються в одну і результат записується в тимчасову послідовність. Потім з основної послідовності вибераються наступні дві відсортовані підпослідовності і так до тих пір доки основна послідовність не стане порожньою. Після цього послідовність з тимчасової переміщається в основну. І знову продовжується сортування злиттям двох відсортованих підпослідовностей. Сортування триватиме до тих пір, поки довжина відсортованої підпослідовності не стане рівною довжині самої послідовності.

Сортування злиттям можна задати рекурсивно: масив поділяється на дві приблизно рівні частини, які після сортування (тим самим способом) зливаються. Коли ж довжина масиву зменшується до 1, відбувається повернення з рекурсії.

Час роботи алгоритму T(n) по впорядкуванню n елементів задовільняє рекурентному співвідношенню: T(n) = 2∙T(𝑛) + O(n), де T(𝑛) — час на

впорядкування половини масиву, O(n) — час на злиття цих половинок. Враховуючи, що T(1) = O(1), розв’язком співвідношення є: T(n) = O(n∙log(n)). Крім того, алгоритм потребує для своєї роботи E(n) додаткової пам’яті. Алгоритм не міняє порядок розташування однакових елементів, а отже він є

стабільним.  
Швидкість алгоритму є асимптотично оптимальною. Але її можна

пришвидшити в константну кількість разів. Можливі оптимізаці:

* •  Оптимізація впорядкування невеликих частин масиву — невеликі частини

масиву (наприклад, при n < 50) впорядковувати сортуванням вставкою.

* •  Оптимізація кількості копіювань елементів — при злитті двох впорядкованих масивів в один, кожен елемент копіюється два рази (спочатку у тимчасовий масив, а потім знову у початковий). Кількість копіювань можна зменшити удвічі, якщо по черзі використовувати для

об’єднання початковий і тимчасовий масиви.  
Нехай маємо дві відсортовані множини X і У; потрібно об’єднати їх у

множину Z , яка також повинна бути відсортованою. У ролі z1 приймаємо min (x1, y1) :якщо z1 = x1, тоді z2 = min(x2, y1 ) і т.д.

ЗАВДАННЯ

1. Ознайомитись з теоретичним матеріалом.  
2. В обраному середовищі програмування створити віконний проект та написати програму, яка реалізує алгоритм сортування злиттям.

ХІД ВИКОНАННЯ

Покроковий опис роботи алгоритму на етапі злиття

Алгоритм M

ДаноX={x1 ,...,xn };Y={y1 ,...,ym }.і -індексдлямножини X, j -індекс для множини Y , k – індекс для множини Z, i=1..n; j=1..m; k=1..(n+m).

M1. Ініціалізація індексів i=1, j=1, k=1;  
M2. Виконувати M3, M4 доки k<n+m.  
M3 [Якщо xi < yi , то [ zk = xi; i=i+1], інакше [ zk=yj, j=j+1]

M4. k=k+1  
M5. Кінець. Вихід.

void **merge**(double \*arr, int low, int high, int mid, int size)

{

int i, j, k;

double c[50];

i = low;

k = low;

j = mid + 1;

*while* (i <= mid && j <= high) {

*if* (arr[i] < arr[j]) {

c[k] = arr[i];

k++;

i++;

}

*else* {

c[k] = arr[j];

k++;

j++;

}

}

*while* (i <= mid) {

c[k] = arr[i];

k++;

i++;

}

*while* (j <= high) {

c[k] = arr[j];

k++;

j++;

}

*for* (i = low; i < k; i++) {

arr[i] = c[i];

}

*for*(int i =0; i<size; ++i)

{

output += QString::number(arr[i], 'g', 3)+" ";

}

output +="\n";

}

void **merge\_sort**(double \*arr, int low, int high, int arrSize)

{

int mid;

*if* (low < high){

*//divide* *the* *array* *at* *mid* *and* *sort* *independently* *using* *merge* *sort*

mid=(low+high)/2;

merge\_sort(arr,low,mid, arrSize);

merge\_sort(arr,mid+1,high, arrSize);

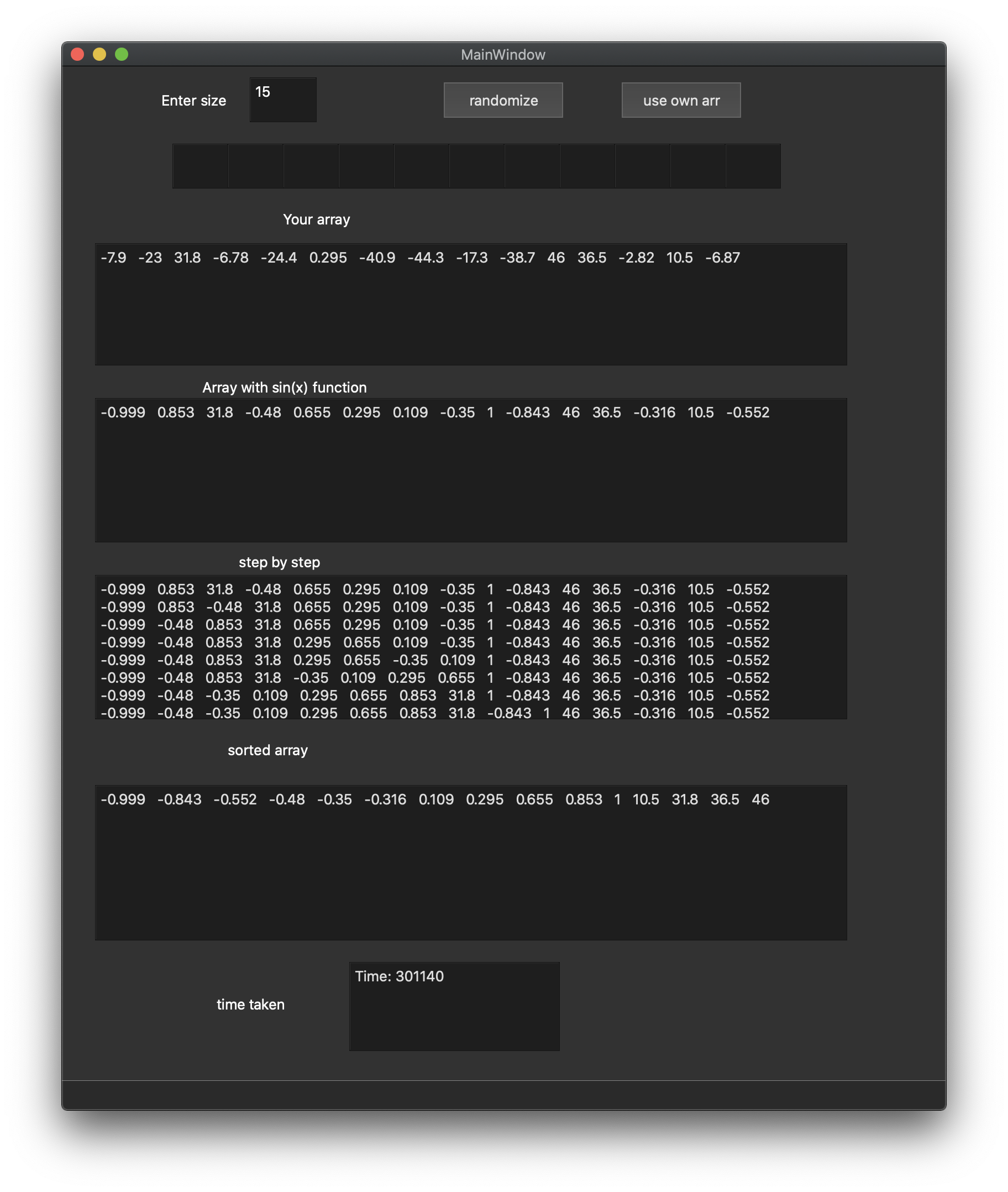
*//merge* *or* *conquer* *sorted* *arrays*

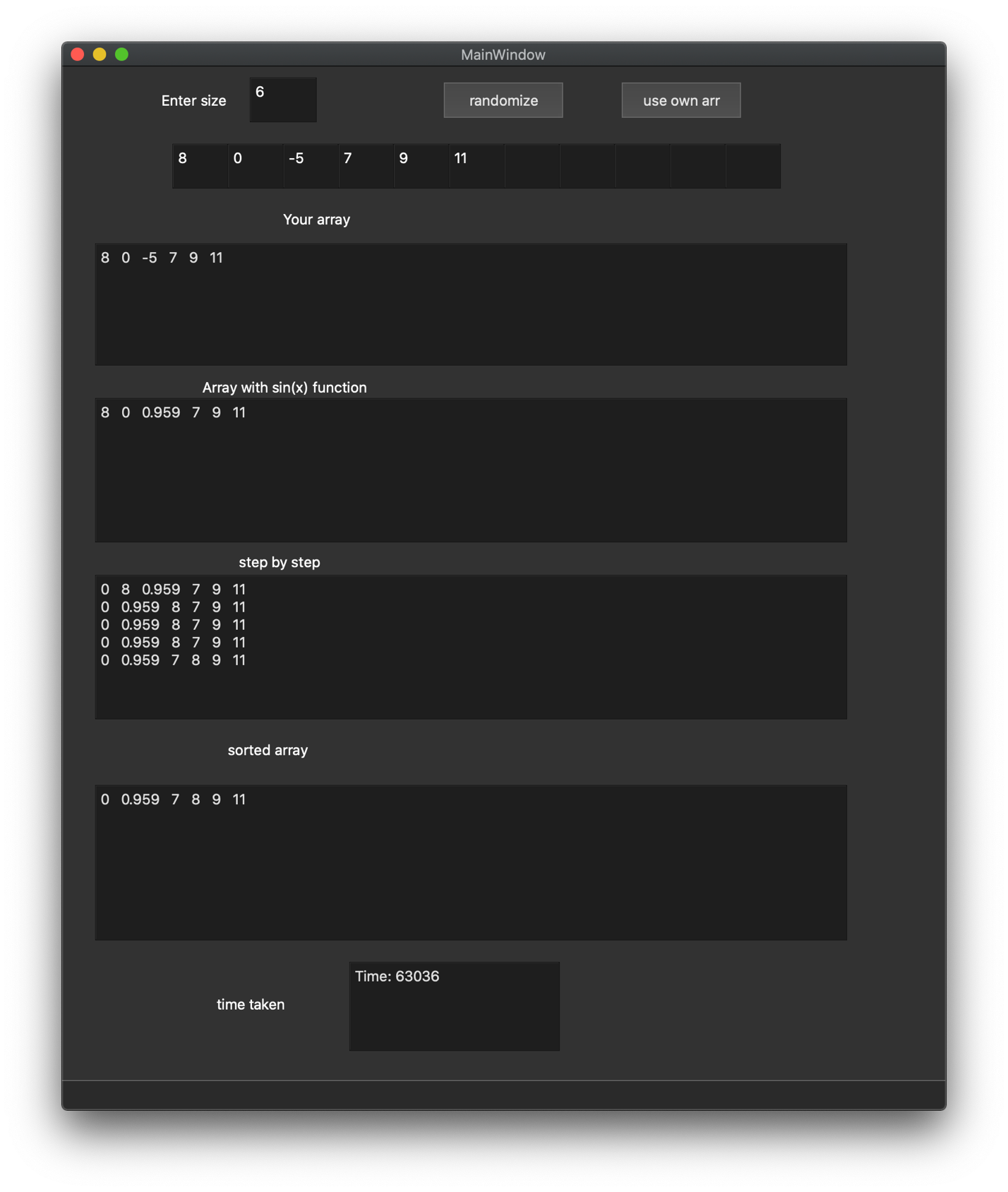
merge(arr,low,high,mid, arrSize);

}

}

**Результати**





ВИСНОВКИ

Я вивчив як застосовувати на практиці алгоритм сортування злиттям. Його суть полягає в методі “розділяй та пануй”. Час роботи алгоритму злиття T(n) для n елементів задовільняє рекурентне співвідношення 𝑇(𝑛) = 2T(n/2) + О(n), де T(n/2) – час на впорядкування половини масиву, О(n) – час на злиття цих половинок. Враховуючи, що Т(1) = О(1), розв’язком співвідношення є: 𝑇(𝑛) = 𝑂(𝑛 ∗ log(𝑛)).

5 3 2 6 4 1 3 7

**3 5** 2 6 **1 4** 3 7

3 5

2 6

2 3 5 6

1 4

3 7

1 3 4 7

*2 3 5 6 1 3 4 7*

2 3 5 6

1 3 4 7

1 2 3 3 4 5 6 7