Т2Ч2

1.

а)

```
найти_макс(массив, начало, конец)
      середина =  (конец-начало)/2
       левый_макс = найти_макс(массив,начало,середина)
       правый_макс = найти макс(масстив, середина+1, конец)
       вернуть левый_макс >= правый макс ? левый макс : правый макс
```

б) Если есть несколько равных элементов, вернется тот из них, что будет левее.
в) $C(n) = \sum_{i=1}^{n-1} 1 = n - 1$
г)
Метод грубой силы предполагает взятие первого элемента за максимум и сравнение его с каждым.
Количество операций при этом так же равно: $C(n) = \sum_{i=1}^{n-1} 1 = n - 1$
Эффективность: O(n)
Эффективность рекурсивного метода: O(n), так как в любом случае надо проверить все элементы массива

Исходя из этого оба алгоритма работают с одинаковой эффективностью, количество операций равное.

2.

а)

```
возведение_в_степень(число, степень)
    если (степень == 0) -> вернуть 1
    если (степень == 1) -> вернуть число
    если (степень четная):
        t = возведение_в_степень(число,степень/2)
        вернуть t
    иначе:
        t = возведение_в_степень(число,(степень-1)/2)
        вернуть число * t * t
```

б)
$C(n) = C\left(\frac{n}{2}\right) + 1$
в)

Критерий	Декомпозиция	Грубая сила
Сложность	O(log n)	O(n)
Кол-во унможений	4 (при n = 16)	16 (при n = 16)
Эффективность	Высокая	Низкая

3.

```
Алгоритм БлижайшаяПара1D(Массив)
    Вход: Массив из n действительных чисел
    Выход: Пара ближайших чисел

    Отсортировать Массив по возрастанию

    МинРасстояние ← бесконечность
    Для i от 1 до Длина(Массив) − 1
        Расстояние ← |Массив[i] − Массив[i − 1]|
        Если Расстояние < МинРасстояние тогда
            МинРасстояние ← Расстояние
            Пара ← (Массив[i − 1], Массив[i])
    Вернуть Пара

```

Класс эффективности:

- Сортировка: $O(n \log n)$
- Линейный проход: $O(n)$
- **Общий класс**: $O(n \log n)$

4.

```
Алгоритм БлижайшаяПара3D(Точки)
    Вход: Массив Точек в 3D-пространстве
    Выход: Пара ближайших точек

    Отсортировать Точки по координате X → ТочкиX
    Отсортировать Точки по координате Y → ТочкиY
    Отсортировать Точки по координате Z → ТочкиZ

    Вернуть РекурсивныйПоиск(ТочкиX, ТочкиY, ТочкиZ)

Алгоритм РекурсивныйПоиск(ТочкиX, ТочкиY, ТочкиZ)
```

```
    Если Количество(ТочкиX) ≤ 3 тогда
        Вернуть ТочнаяПроверка(ТочкиX)

    Середина ← Длина(ТочкиX) / 2
    ЛеваяX ← ТочкиX[0 : Середина]
    ПраваяX ← ТочкиX[Середина : Конец]
    X_граница ← ТочкиX[Середина].X

    ЛеваяY ← Точки из ТочкиY, где X ≤ X_граница
    ПраваяY ← Остальные из ТочкиY

    ЛеваяZ ← Точки из ТочкиZ, где X ≤ X_граница
    ПраваяZ ← Остальные из ТочкиZ

    (Пара1) ← РекурсивныйПоиск(ЛеваяX, ЛеваяY, ЛеваяZ)
    (Пара2) ← РекурсивныйПоиск(ПраваяX, ПраваяY, ПраваяZ)

    δ ← min(Расстояние(Пара1), Расстояние(Пара2))

    Пара3 ← БлижайшиеВСлое(ТочкиY, δ, X_граница)

    Вернуть пару с минимальным расстоянием из (Пара1, Пара2, Пара3)

Алгоритм БлижайшиеВСлое(ТочкиY, δ, X_граница)
    Слой ← точки из ТочкиY, где |X − X_граница| < δ

    МинРасстояние ← δ
    Пара ← пусто

    Для i от 1 до Длина(Слой)
        Для j от i+1 до min(i+15, Длина(Слой))
            Если Расстояние(Слой[i], Слой[j]) < МинРасстояние тогда
                МинРасстояние ← Расстояние(Слой[i], Слой[j])
                Пара ← (Слой[i], Слой[j])
    Вернуть Пара

```

Класс эффективности:

- Рекурсивная декомпозиция: $2T(n/2)$
- Обработка слоя: $O(n)$
- **Итоговая сложность**: $O(n \log n)$