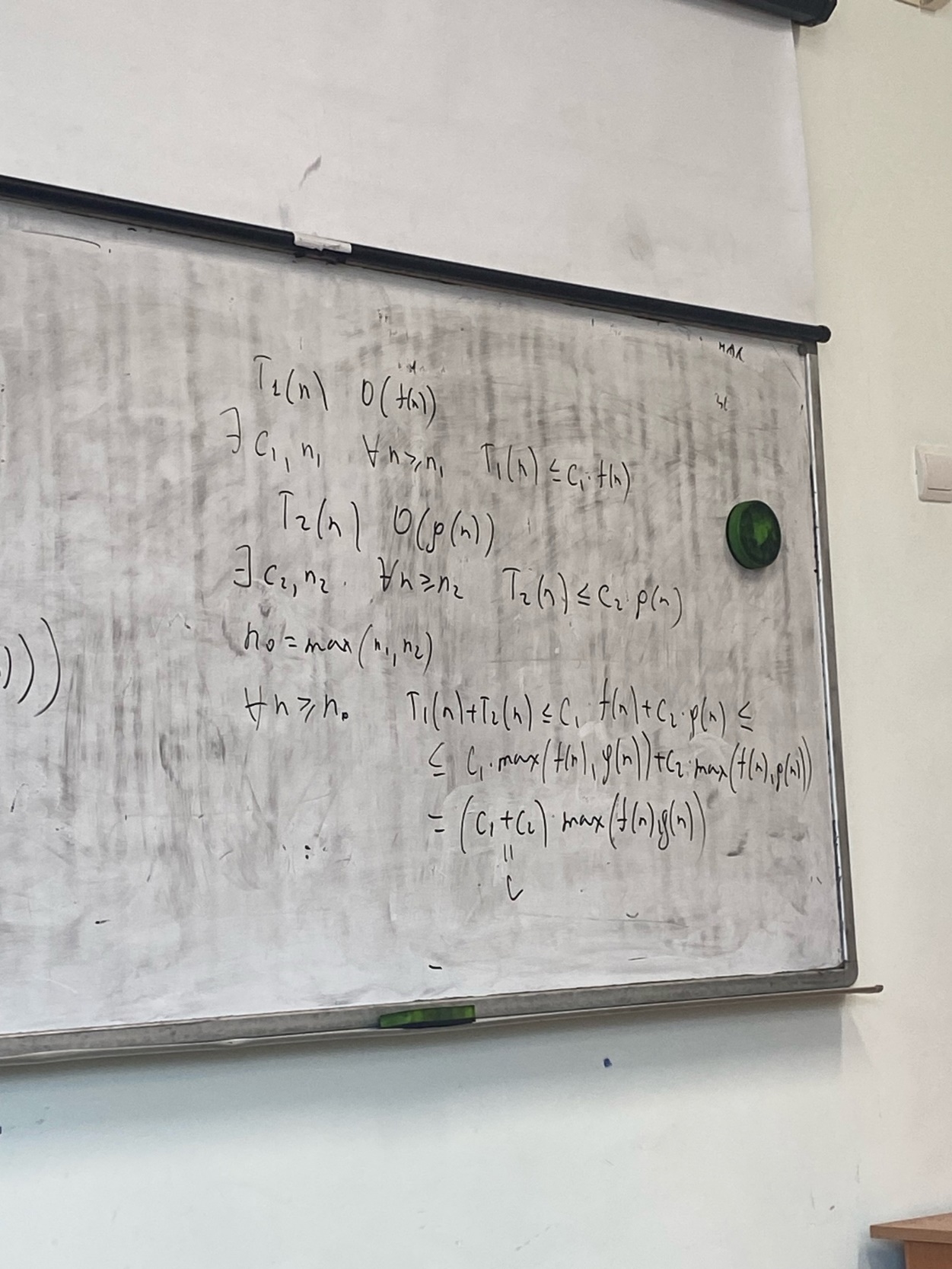
**Лекция 26.02.2024  
Тема: Вычисление времени выполнения программы (Сортировка пузырьком)**

**Лемма «Правило суммы»**Пусть T1(n) – время выполнения программы P1 и имеет степень роста O(f(n))  
T2(n) – время выполнения программы P2 и имеет степень роста O(g(n))  
Тогда время последовательного выполнения программ = T1(n) + T2(n) и имеет степень роста O(max(f(n), g(n)))  
Время выполнения конечной последовательности программных фрагментов без учета констант имеет порядок фрагмента с наибольшим временем выполнения

Замечание:  
В общем случае нахождение времени выполнения программы (верхняя граница) задача сложная. Не существует полного набора правила анализа программ. Только некоторые из них базовые.

**Базовые правила анализа программ:**

1. Время выполнения операторов присваивания ввода/вывода обычно имеет порядок O(1), но есть исключения, например в языках, где можно присваивать большие массивы или в случае вызова функции.
2. Время выполнения последовательности операторов вычисляется с помощью правила сумм, как наибольшее из времени выполнения операторов данной последовательности.
3. Оператор ветвления (условный оператор) имеет вид [если \*логическое выражение\* то \*оператор 1\* иначе \*оператор 2\*]. Время вычисление времени логического выражения обычно O(1), тогда время выполнения всей конструкции это max(оператор 1, оператор 2).
4. Время выполнения цикла вычисляется как сумма времени выполнения всех итераций цикла (1 итерация – время проверки условия). Часто время выполнения цикла вычисляется как произведение количество итераций цикла на наибольшее возможное время выполнения тела цикла для 1 итерации.

**Алгоритмы сортировки**Сортировка пузырьком (Bubble Sort)  
  
(1) for i = 1 to n – 1 do:  
(2) for j = n downto i + 1 do:  
(3) if a[j – 1] > a[j] then:  
 begin  
(4) temp = a[j – 1];  
(5) a[j – 1] = a[j];  
(6) a[j] = temp;  
 end.  
  
Каждый из операторов (4), (5), (6) выполняется за О(1), тогда время выполнения программного фрагмента (4) – (6) вычисляется по формуле О(max(1, 1, 1)) = O(1)  
(3) – (6) складывается из проверки условия О(1) и ветка (4) – (6), время выполнения которой О(1). Следовательно (3) – (6) тоже О(1).  
Цикл (2) – (6) оценивается как О((n - i) \* 1) = O(n - i) (1 – время выполнения 1 итерации,   
n – i – количество итераций)  
Цикл (1) – (6) оценивается как = => O(n\*\*2)

ДЗ в тг