Завдання для самостійної роботи

2.7. Нехай $f(n) = 3n^2 - n + 4$ та $g(n) = n\log n + 5$. Покажіть що

$$f(n) + g(n) = O(n^2).$$

2.8. Нехай $f(n) = \sqrt{n}$ та $g(n) = \log n$. Покажіть що

$$f(n) + g(n) = O(\sqrt{n}).$$

2.7 $f(n) = 3n^2 - n + 4$ $\quad g(n) = n\log(n) + 5$

$f(n) + g(n) = 3n^2 + n\log(n) - n + 9 \stackrel{?}{=} O(n^2)$

I) приберемо сталу 9

$3n^2 + n\log(n) - n$

II) розглянемо функцію $j(n) = 4n^2$

$3n^2 + n\log(n) - n = 4n^2$

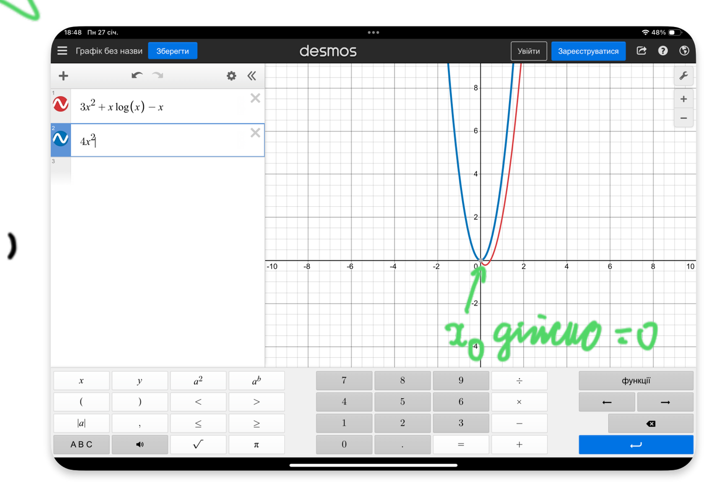
$-n^2 + n\log(n) - n = 0$

$n = 0 + \Delta \quad \forall\Delta > 0$

почнем шукати $x_0 = 0 + \Delta \quad \forall\Delta > 0$

Отже $f(n) + g(n) \leq O(4n^2) \sim O(n^2)$, при чому $x_0 = 0$

proof



x_0 рівно $= 0$

P.S. Ми можем відкинути доданки $n\log(n)$ та $-n$, адже вони зростають повільніше, ніж $3x^2$, а очевидно що $4x^2 \geq 3x^2$

№ 2.11

2.11. Нехай f(n) функція визначена у вправі 2.10. Розглянемо функцію

```
def g(n):
    sum = 0
    for i in range(1, n + 1):
        sum = sum + i + f(i)
    return sum
```

Визначте асимптотичну оцінку виконання функції g(n) у найгіршому випадку в термінах O–«великого». Що є результатом виконання функції g(n) для заданого натурального числа n. Чи можна оптимізувати цю функцію, покращивши її асимптотичну оцінку?

12

```
def g(n):
    sum = 0                        | O(1)
    for i in range(1, n + 1):        O(n)
        sum = sum + i + f(i)         O(n^2) = ΣΣ
    return sum                     O(1)
```

Результат $O(n^2)$

На 2 останні питання отримаємо відповідь розписавши

Можно $\sum_{i=1}^{n} i + \sum_{j=1}^{i} i = \sum_{i=1}^{n} i + \frac{i(i+1)}{2} =$

$= \frac{i(i+1)}{2} + \frac{1}{2}\sum_{i=1}^{n}(i^2 + i) = \frac{n(n+1)}{2} + \frac{n(n+1)}{4} + \frac{n(n+1)(2n+1)}{12} =$

$= \frac{n^3 + 6n^2 + 5n}{6}$

```
def g(n):
    sum = (n**3 + 6*n**2 + 5*n) // 6    | O(1)
    return sum    | O(1)
```

№ 2.10

2.10. Визначте асимптотичну оцінку виконання функції у найгіршому випадку в термінах O–«великого» для функції:

```
def f(n):
    sum = 0
    for i in range(1, n + 1):
        sum = sum + i
    return sum
```

Що є результатом виконання наведеної функції для заданого натурального числа n? Чи можна оптимізувати цю функцію, покращивши її асимптотичну оцінку?

```
def f(n):
    sum = 0                      O(1)
    for i in range(1, n + 1):    O(n)
        sum = sum + i            O(n)
    return sum                   O(1)
```

$2\,O(1) + 2\,O(n) = O(n) \leftarrow$ найбільше

Результатом – Σ арифметичної прогресії від 1 до n

Покращення:

```
def bobs(n):
    sum = (1+n)*n // 2    | O(1)
    return sum    | O(1)
```

результат : $O(1)$

№ 2.12

2.12. Нехай f(n) функція визначена у вправі 2.10, а функція g(n) – у вправі 2.11. Розглянемо функцію

```
def h(n):
    return f(n) + g(n)
```

Визначте асимптотичну оцінку виконання функції h(n) у найгіршому випадку в термінах O–«великого». Що є результатом її виконання для заданого натурального числа n. Чи можна оптимізувати цю функцію, покращивши її асимптотичну оцінку?

Очевидно, що $O(n) + O(n^2) = O(n^2)$

Очевидно можна покращити до $O(1)$

```
def h(n):
    sum = (n + 1)*n // 2 + (n**3 + 6*n**2 + 5*n) // 6    O(1)
    return sum    O(1)
```

2.13

2.13. Знайдіть асимптотичний час виконання програми у явному вигляді, якщо для нього відоме рекурентне співвідношення

a) $T(n) = \begin{cases} O(1), & n = 0; \\ T(n-1) + O(1), & n \geq 1. \end{cases}$

b) $T(n) = \begin{cases} O(1), & n \leq a,\ a > 1; \\ T(n-a) + O(1), & n > a. \end{cases}$

c) $T(n) = \begin{cases} O(1), & n = 0; \\ aT(n-1) + O(1), & n \geq 1, a > 1. \end{cases}$

d) $T(n) = \begin{cases} O(1), & n \leq a,\ a > 1; \\ aT(n-a) + O(1), & n > a. \end{cases}$

e) $T(n) = \begin{cases} O(1), & n = 0; \\ T(n-1) + O(n), & n \geq 1. \end{cases}$

f) $T(n) = \begin{cases} O(1), & n \leq a,\ a > 1; \\ T(n-a) + O(n), & n > a. \end{cases}$

g) $T(n) = \begin{cases} O(1), & n = 1; \\ aT([n/a]) + O(1), & n \geq 2, a \geq 2. \end{cases}$

h) $T(n) = \begin{cases} O(1), & n = 1; \\ aT([n/a]) + O(n), & n \geq 2, a \geq 2. \end{cases}$

a) $T(n-1) + C \leq (T(n-2) + C) + C =$

$= T(n-2) + C(1+1) = T(n-2) + C(2) \leq$

$\leq T(n-n) + C(n) \sim O(n)$

d) $aT(n-a) + C \leq a(aT(n-2a) + C) + C =$

$= a^2T(n-2a) + aC + C = a^2T(n-2a) +$

$+ C(a+1) = a^2T(n-2a) + C(a^2-1) \leq$

$\leq a^2(aT(n-3a) + C) + C(a^2-1) =$

$= a^3T(n-3a) + C(a^3-1) \leq \ldots$

$\ldots \leq a^{[\frac{n}{a}]} + C(a^{[\frac{n}{a}]} - 1) = O(a^n)$

g) $aT([n/a]) + C \leq$

$\leq a(aT([\frac{n}{a^2}]) + C) + C =$

$= a^2T([\frac{n}{a^2}]) + aC + C =$

$= a^2T([\frac{n}{a^2}]) + C(a^2-1) \leq$

$\leq |\ \text{let } \log_a n = m\ | \leq$

$\leq a^m T([\frac{n}{a^m}]) + C(a^m - 1) =$

$= a^m C + C(a^m - 1) = O(n)$

P.S. $a^{\log_a n} = n$

h) Як і в g, але

Отримаємо

$\leq a^m C + Cn \cdot (a^m - 1) =$

$= Cn + Cn^2 = O(n^2)$