ТЕМА 8. РЕКУРЕНТНІ СПІВВІДНОШЕННЯ

Рекурентні співвідношення

Нехай $\{a_n : n \geq 0\}$ деяка послідовність дійсних чисел.

Рекурентне співвідношення 1-го порядку

послідовність $\{a_n : n \geq 0\}$ задана *рекурентним співвідношенням першого порядку*, якщо явно задано її перший член a_0, а кожен наступний член a_n цієї послідовності визначається деякою залежністю через її попередній член a_{n-1}, тобто

$$\begin{cases} a_0 = a \\ a_n = f(n, p, a_{n-1}), n \geq 1 \end{cases}$$

де a числове значення, p – деякий сталий параметр або набір параметрів, f – функція, задана аналітично у вигляді арифметичного виразу, що складається з операцій доступних для виконання з допомогою Python

Розглянемо послідовність $\{a_n = n! : n \geq 0\}$
Її можна задати рекурентним співвідношенням першого порядку

$$\begin{cases} a_0 = 1 \\ a_n = n a_{n-1}, n \geq 1 \end{cases}$$

Наприклад, якщо потрібно знайти a_5, то використовуючи рекурентні формули, послідовно від першого члена отримаємо

$$a_0 = 1$$
$$a_1 = 1 \cdot a_0 = 1 \cdot 1 = 1$$
$$a_2 = 2 \cdot a_1 = 2 \cdot 1 = 2$$
$$a_3 = 3 \cdot a_2 = 3 \cdot 2 = 6$$
$$a_4 = 4 \cdot a_3 = 4 \cdot 6 = 24$$
$$a_5 = 5 \cdot a_4 = 5 \cdot 24 = 120$$

Для обчислення членів послідовностей, заданих рекурентними співвідношеннями, використовують цикли.

Нехай послідовність a_n задана рекурентним співвідношенням

$$\begin{cases} a_0 = a \\ a_n = f(n, p, a_{n-1}), n \geq 1 \end{cases}$$

Тоді, після виконання коду

```
x = a
for n in range(1, N+1):
    x = f(n, p, x)
```

у змінній `x` буде міститися значення елемента a_N послідовності

Для введеного з клавіатури значення N обчислити N!

Як було зазначено раніше послідовність $a_n = n!$ може бути задана рекурентним співвідношенням

$$\begin{cases} \boxed{a_0 = 1} \\ \boxed{a_n = n a_{n-1},} \boxed{n \geq 1} \end{cases}$$

N!

```
N = int(input("N = "))
x = 1                       # x = a
for n in range(1, N+1):
    x = n*x                 # x = f(n, p, x)
print (N, "! = ", x)
```

Скласти програму для обчислення елементів послідовності

$$a_n = \frac{x^n}{n!}, n \geq 0.$$

Складемо рекурентне співвідношення для заданої послідовності.

Легко бачити, що кожен член послідовності a_n є добутком чисел. Враховуючи це, обчислимо частку двох сусідніх членів послідовності. Для $n \geq 1$ отримаємо

$$\frac{a_n}{a_{n-1}} =$$

$$= \frac{x^n}{n!} \cdot \frac{(n-1)!}{x^{n-1}} =$$

$$= \frac{x}{n}$$

Звідки випливає, що для $n \geq 1$

$$a_n = \frac{x}{n} a_{n-1}$$

Отже ми отримали для послідовності a_n рекурентну формулу, у якій кожен член послідовності для $n \geq 1$ визначається через попередній член a_{n-1}.
Щоб задати рекурентне співвідношення, залишилося задати перший член a_0

$$a_0 = \frac{x^0}{0!} = 1.$$

Отже остаточно

$$\begin{cases} a_0 = 1 \\ a_n = \dfrac{x}{n} a_{n-1}, n \geq 1 \end{cases}$$

$$\begin{cases} a_0 = 1 \\ a_n = \dfrac{x}{n} a_{n-1}, n \geq 1 \end{cases}$$

Рекурентні співвідношення

```
N = int(input("N = "))
x = float(input("x = "))
a = 1
for n in range(1, N+1):
    a = x / n * a
print ("a = ", a)
```

Скласти програму для обчислення суми

$$S_n = \sum_{i=1}^{n} 2^{n-i} i^2 \,, n \geq 1.$$

Складемо рекурентне співвідношення для заданої послідовності.

Підставляючи $n = 1$, отримаємо $S_1 = 1$.

Щоб отримати вираз для загального члена, розкриємо суму

$$S_n = 2^n \left(\frac{1^2}{2^1} + \frac{2^2}{2^2} + \ldots + \frac{(n-1)^2}{2^{n-1}} + \frac{n^2}{2^n} \right) =$$

$$= 2 \cdot 2^{n-1} \left(\frac{1^2}{2^1} + \frac{2^2}{2^2} + \ldots + \frac{(n-1)^2}{2^{n-1}} + \frac{n^2}{2^n} \right) =$$

$$2 \cdot 2^{n-1} \left(\frac{1^2}{2^1} + \frac{2^2}{2^2} + \ldots + \frac{(n-1)^2}{2^{n-1}} \right) + 2 \cdot 2^{n-1} \frac{n^2}{2^n} =$$

$$= 2 \cdot S_{n-1} + n^2$$

Отже, рекурентне співвідношення для буде мати вигляд

$$\begin{cases} S_1 = 1, \\ S_n = 2 \cdot S_{n-1} + n^2, n \geq 2. \end{cases}$$

Підрахунок суми

```
N = int(input("N = "))
S = 1
for n in range(2, N+1):
    S = 2 * S + n ** 2
print ("S = ", S)
```

Скласти програму для обчислення суми

$$S_n = \sum_{i=0}^{n} a^i b^{n-i}$$

Співвідношення старших порядків

послідовність $\{a_n : n \geq 0\}$ задана *рекурентним співвідношенням m-го порядку*, якщо

$$\begin{cases} a_0 = a, a_1 = b, \dots, a_{m-1} = c, \\ a_n = f(n, p, a_{n-1}, \dots, a_{n-m},), n \geq m \end{cases}$$

де $a, b, \dots, c$ – числові сталі, *p* – деякий сталий параметр або набір параметрів, *f* – функція, задана аналітично.

Послідовність $\{F_n : n \geq 0\}$ чисел Фібоначі задають рекурентрим співвідношенням другого порядку

$$\begin{cases} F_0 = 1, F_1 = 1 \\ F_n = F_{n-1} + F_{n-2}, n \geq 2 \end{cases}$$

$F_0 = 1, F_1 = 1$

$F_2 = F_1 + F_0 = 1 + 1 = 2$

$F_3 = F_2 + F_1 = 2 + 1 = 3$

$F_4 = F_3 + F_2 = 3 + 2 = 5$

$F_5 = F_4 + F_3 = 5 + 3 = 8$

$F_6 = F_5 + F_4 = 5 + 3 = 13$

Для обчислення елементів послідовності, заданої рекурентним співвідношенням вищого порядку, застосовується інший підхід ніж для співвідношень першого порядку.

Алгоритм наведемо на прикладі співвідношення 3-го порядку

Нехай послідовність a_n задана рекурентним співвідношенням

$$\begin{cases} a_0 = a, a_1 = b, a_2 = c, \\ a_n = f(n, p, a_{n-1}, , a_{n-2}, a_{n-3},), n \geq 3 \end{cases}$$

Тоді, після виконання коду

```
x3 = a    # x3 - змінна для (n-3)-го члену послідовності
x2 = b    # x2 - змінна для (n-2)-го члену послідовності
x1 = c    # x1 - змінна для (n-1)-го члену послідовності
for n in range(3, N+1):
    x = f(n, p, x1, x2, x3) # Обчислення наступного члену
    x3 = x2  # Зміщення змінних для наступних ітерацій
    x2 = x1
    x1 = x
```

у змінних x і x1 буде міститися a_N, у змінній x2 – a_{N-1} , а в змінній x3 – a_{N-2}.

Знайти N-й член послідовності Фібоначі

Як було зазначено раніше послідовність чисел Фібоначі F_n може бути задана рекурентним співвідношенням

$$\begin{cases} \boxed{F_0 = 1, F_1 = 1} \\ \boxed{F_n = F_{n-1} + F_{n-2}}, \boxed{n \geq 2} \end{cases}$$

Числа Фібоначі

```
N = int(input("N = "))
F2 = 1; F1 = 1
for n in range(2, N+1):
    F = F1 + F2
    F2 = F1
    F1 = F
print ("F = ", F)
```

Знайдіть N-й член послідовності заданої рекурентним співвідношенням

$$\begin{cases} a_0 = a_1 = a_2 = 1 \\ a_n = a_{n-1} + a_{n-3}, n \geq 3 \end{cases}$$

Вищенаведена теорія легко узагальнюється на системи рекурентних співвідношень, якщо вважати, що послідовності є векторними.

Вивчимо системи рекурентних співвідношень на прикладах

Скласти програму для обчислення суми

$$S_n = \sum_{i=0}^{n} \frac{x^i}{i!}, \, , n \geq 0.$$

Розкриваючи суму побачимо, що

$$S_n = \boxed{1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots + \frac{x^{n-1}}{(n-1)!}} + \frac{x^n}{n!} =$$

$$= S_{n-1} + \frac{x^n}{n!}$$

Отже послідовність S_n визначається рекурентним співвідношенням

$$\begin{cases} S_0 = 1, \\ S_n = S_{n-1} + \dfrac{x^n}{n!}, n \geq 1 \end{cases}$$

Позначаючи

$$a_n := \frac{x^n}{n!}, n \geq 0.$$

і згадуючи, що ми вже для цієї послідовності раніше отримали рекурентне співвідношення, запишемо систему рекурентних співвідношень для S_n

$$\begin{cases} S_0 = 1, a_0 = 1 \\ a_n = \dfrac{x}{n} a_{n-1}, n \geq 1, \\ S_n = S_{n-1} + a_n, n \geq 1. \end{cases}$$

Підрахунок суми

```
N = int(input("N = "))
x = float(input("x = "))
a = 1
S = 1
for n in range(1, N+1):
    a = x / n * a
    S = S + a
print ("S = ", S)
```

Обчислити суму, задану рекурентним співвідношенням

$$S_n = \sum_{k=0}^{n} \frac{a_k}{2^k},$$

де $a_0 = a_1 = a_2 = 1, a_k = a_{k-1} + a_{k-3}, k \geq 3.$

Досі ми будували програми, що знаходять значення члену послідовності за його номером.
Проте, часто постає задача, коли потрібно знайти найперший член послідовності, що задовольняє певну умову.
У такому разі цикл **`for`** заміняється циклом з умовою продовження **`while`**.

Послідовність задана рекурентним співвідношенням

$$\begin{cases} x_0 = 1, x_1 = 0, x_2 = 1, \\ x_n = x_{n-1} + 2x_{n-2} + x_{n-3}, n \geq 3. \end{cases}$$

Створити програму для знаходження номера найменшого члена цієї послідовності, який перевищує число a.

Нехай змінна `x` пробігає послідовність x_n. Тоді умовою завершення циклу є умова `x > a`, а, значить, цикл має виконуватися поки виконується умова `x <= a`.

Співвідношення з умовою

```
a = int(input("a = "))
x3 = 1
x2 = 0
x1 = 1
x = x1 # Змінна x використовується в умові циклу,
       # тому вона має бути оголошена до початку циклу
n = 2  # Номер елементу, що міститься у змінній x
while x <= a:
    n += 1 # У змінній n міститься поточний
           # номер елементу послідовності
    x = x1 + 2*x2 + x3
    x3 = x2
    x2 = x1
    x1 = x

print ("x(",n,") = ", x)
```

Наближене обчислення e^x.

Відомо, що функція e^x обчислюється як сума ряду

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} = \boxed{1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots + \boxed{\frac{x^n}{n!}}} + \cdots$$

Позначимо загальний член ряду $\boxed{a_n}$, а часткову суму

$$\boxed{S_n = \sum_{i=0}^{n} \frac{x^i}{i!},}$$

Очевидно, що $S_n \to e^x, n \to \infty$.

Під наближеним (з точністю ε) значенням границі послідовності S_n будемо розуміти такий член S_N , що виконується співвідношення

$$|S_N - S_{N-1}| = |a_N| < \varepsilon$$

Раніше було отримано, що послідовність S_n визначається системою рекурентних співвідношень

$$\begin{cases} S_0 = 1, a_0 = 1 \\ a_n = \frac{x}{n} a_{n-1}, n \geq 1, \\ S_n = S_{n-1} + a_n, n \geq 1. \end{cases}$$

Наближене обчислення експоненти

```
from math import fabs # імпортуємо функцію для обчислення
                      # модуля дійсного числа
eps = 0.000001        # Точність обчислення
x = float(input("x = "))
S = a = 1
n = 0
while fabs(a) >= eps:
    n += 1
    a *= x / float(n)
    S += a
print ("exp(",x,") = ", S)
```