Отчёт ИПР1

Выполнил: Заломов Роман Андреевич, 121702

Листинг программы для вычисления рядов Тейлора для синуса. Язык программирования - Python:

```
import math


DELTA = 1E-4
PI = math.pi


def tailor_term(n: int, x: float) -> float:
    return (-1)**n*((x**(2*n+1))/math.factorial(2*n+1))


def sin_tailor(term_num: int, x: float):
    # while x >= 2*PI:
    #     x -= 2*PI
    tailor_list = [tailor_term(i, x) for i in range(term_num)]
    # print("Tailor terms:", ', '.join(list(map(str, tailor_list))))
    sin = sum(tailor_list)
    return sin


def scores() -> str:
    return "-"*200


print(scores())
print("Computed sin\t\tTailor sin")
for x in range(30):
    print(scores())
    print(f"Computing sin({x})")
    n = 1
    while abs(math.sin(x) - sin_tailor(n, x)) >= DELTA:
        print(f"{n}. Computed sin({x}): {math.sin(x)}\t\tTailor sin({x}):
{sin_tailor(n, x)}\t\t"
              f"E={abs((math.sin(x)-sin_tailor(n, x))/sin_tailor(n, x))}"
              f"\t\tdelta={abs(math.sin(x) - sin_tailor(n, x))}")
        n += 1
    try:
        print(f"{n}. Computed sin({x}): {math.sin(x)}\t\tTailor sin({x}):
{sin_tailor(n, x)}\t\t"
              f"E={abs((math.sin(x) - sin_tailor(n, x)) / sin_tailor(n, x))}"
              f"\t\tdelta={abs(math.sin(x) - sin_tailor(n, x))}")
    except ZeroDivisionError:
        print(f"{n}. Computed sin({x}): {math.sin(x)}\t\tTailor sin({x}):
{sin_tailor(n, x)}\t\t"
              f"E=0\t\tdelta={abs(math.sin(x) - sin_tailor(n, x))}")
    print(f"{n} tailor terms needed to count sin with {DELTA} accuracy")
```

Для получения заранее просчитанного значения синуса использовался модуль math. Для того, чтобы синус, просчитанный с помощью ряда Тейлора считался правильным, разница между им и заранее просчитанным синусом должна составлять не более 10^-4(константа DELTA в листинге).

Пример вывода:

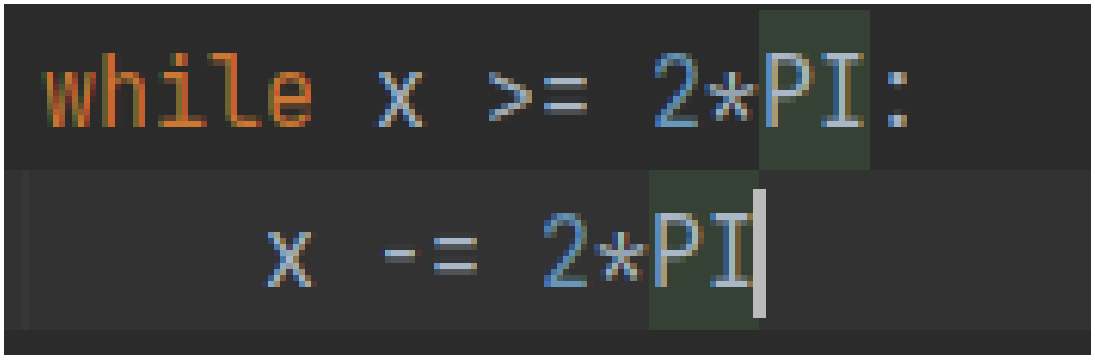
```
Computing sin(9)
1. Computed sin(9): 0.4121184852417566      Tailor sin(9): 9.0      E=0.9542090571953603        delta=8.587881514758243
2. Computed sin(9): 0.4121184852417566      Tailor sin(9): -112.5       E=1.0036632754243713        delta=112.91211848524176
3. Computed sin(9): 0.4121184852417566      Tailor sin(9): 379.575      E=0.9989142633597002        delta=379.1628815147582
4. Computed sin(9): 0.4121184852417566      Tailor sin(9): -569.4267857142856       E=1.0007237427103552        delta=569.8389041995274
5. Computed sin(9): 0.4121184852417566      Tailor sin(9): 498.20022321428587       E=0.9991727854263436        delta=497.7881047290441
6. Computed sin(9): 0.4121184852417566      Tailor sin(9): -287.9614833603895       E=1.0014311583633773        delta=288.37360184563124
7. Computed sin(9): 0.4121184852417566      Tailor sin(9): 120.23786428415349       E=0.9965724733411114        delta=119.82574579891173
8. Computed sin(9): 0.4121184852417566      Tailor sin(9): -37.21045552159879       E=1.0110753410423192        delta=37.622574006840544
9. Computed sin(9): 0.4121184852417566      Tailor sin(9): 9.676727949967152        E=0.957411380440518     delta=9.264609464725394
10. Computed sin(9): 0.4121184852417566     Tailor sin(9): -1.428131293298465       E=1.2885718471233218        delta=1.8402497785402216
11. Computed sin(9): 0.4121184852417566     Tailor sin(9): 0.7135201321884752       E=0.42241505649220257       delta=0.3014016469467186
12. Computed sin(9): 0.4121184852417566     Tailor sin(9): 0.3706866036026211       E=0.11177064732436559       delta=0.04143188163913547
13. Computed sin(9): 0.4121184852417566     Tailor sin(9): 0.4169691299617114       E=0.011633102720098843      delta=0.004850644719954833
14. Computed sin(9): 0.4121184852417566     Tailor sin(9): 0.4116288384587395       E=0.001189534690646284      delta=0.0004896467830171058
15. Computed sin(9): 0.4121184852417566     Tailor sin(9): 0.41216155226630197      E=0.00010449064040196407        delta=4.306702454537348e-05
15 tailor terms needed to count sin with 0.0001 accuracy
```

Computed sin – заранее просчитанный синус
Tailor sin – синус, подсчитанный с помощью рядов Тейлора при очередной итерации
E – относительная погрешность
delta – разница между заранее просчитанным синусом и синусом, подсчитанным при помощи ряда Тейлора.

Как видно, для обеспечения требуемой точности потребовалось просчитать и просуммировать 15 слагаемых ряда (последняя строка).

В данной конфигурации можно просчитать sin(t) до t = 30 (не включая t = 30). Т.е., начиная с t = 30, увеличение количества слагаемых ряда Тейлора не помогает.

Т.к. Python поддерживает только один вещественный тип(float), то для увеличения «радиуса сходимости» требуется использовать формулы приведения, т.е. привести аргумент к виду 0 < t < 2π.



(PI = 3.141592653589793)

Это приведёт к потере точности(ибо проводятся дополнительные вычислительные процедуры). Но синус можно будет вычислять для больших чисел. Также в этом случае можно увеличить точность с 10^-4, до, например, 10^-8. И даже при такой хорошей точности, синус вычисляется для достаточно больших чисел(проверено на числах <= 10000). Так, что, если использовать данный метод подсчёта, то «радиус сходимости» будет «неограниченным», что проверить достаточно сложно. Единственным способом «сломать» метод будет введение достаточно маленького числа погрешности, например, 10^-15, что позволит сократить «радиус сходимости» (при точности 10^-15 максимальное число, для которого вычисляется синус, составляет 4). Но такая точность является «запредельной» и ненужной.