SER: Сумма ряда

Задача: написать программу, выводящую таблицу из M значений некоторой функции $f(x)$ на промежутке от a до b. Функция задана как сумма бесконечного ряда.

$$f(x) = x^2 - \frac{2^3}{4!}x^4 + \frac{2^5}{6!}x^6 - \cdots = \sum_{k=1}^{\infty}(-1)^{k-1}\frac{2^{2k-1}}{(2k)!}x^{2k} = \sum_{k=1}^{\infty}u_k$$

Из математического анализа известно, что при $|x| \leq 1$ эта функция совпадает с $g(x) = \sin^2 x$.

0. **Используйте грубое приближение (0 баллов).** Заведите в программе функцию $f(x) = x^2$ (грубое приближение суммы ряда) и эталонную функцию $g(x) = \sin^2 x$. Положите, для примера, $x = 0.5$ и напечатайте координату x, значения $f(x)$, $g(x)$ и их разность (т.е. ошибку).

   ```
   x = 0.500000  f = 0.250000  g = 0.229849  delta = 0.020151
   ```

1. **Уточните значение функции (+1 балл).** Научитесь вычислять конечные суммы такого вида, возьмите для начала первые `N=3` слагаемых. В будущем будем увеличивать `N` (потенциально до бесконечности), так что делайте сразу в общем виде и правильно. Обратите внимание, что 2^{2k-1} не помещается в `int` при всех $k > 16$, а $(2k)!$ — уже при $k > 6$ (на 32-битных системах). Поэтому:

   – *В тетради проведите вычисления:* выпишите первое слагаемое u_1 и вычислите отношение следующего слагаемого к текущему, т.е. получите множитель $v_k = u_{k+1}/u_k$, который будет уже выражаться простой формулой (почти всё сократится). Имея этот множитель и текущее слагаемое, вы можете вычислить следующее. И следующее. И следующее.

   – *Примените формулы в программе в цикле.* Обратите внимание, несмотря на то, что в формулах переменные содержат индексы, в вашей программе массивы не нужны, достаточно работать с текущим и со следующим слагаемым. Исправьте функцию `f()` так, чтобы она в цикле вычисляла конечное (`N=3`) число слагаемых и возвращала накопленную сумму. Для самоконтроля выводите на экран очередное слагаемое и значение частичной суммы.

   ```
   u =  0.250000  s = 0.250000
   u = -0.020833  s = 0.229167
   u =  0.000694  s = 0.229861

   x = 0.500000  f = 0.229862  g = 0.229849  delta = 0.000013
   ```

2. **Используйте машинное “эпсилон” (+1 балл).** Замените арифметический цикл `for` на цикл `while` — вместо того, чтобы всегда брать фиксированное число слагаемых, продолжайте цикл до тех пор, пока вычисления имеют смысл. В этой сумме слагаемые быстро уменьшаются, логично предположить, что с некоторого момента они окажутся меньше машинного “эпсилон” для уже имеющейся суммы и не смогут её изменить, т.е. $s_k + u_{k+1} = s_k$. Это условие и следует использовать в качестве остановки цикла.

При таких более точных вычислениях разность между нашей и эталонной функцией будет близко к нулю. Для вывода значения разности используйте спецификатор научной формы записи числа `%le`, чтобы контролировать порядок ошибки.

3. **Постройте таблицу значений функции (+1 балл).** Уберите отладочный вывод значений слагаемых, которым пользовались в предыдущих пунктах. Вместо того, чтобы вычислять одно значение в фиксированной точке, вычислите и выведите таблицу из нескольких значений функции на отрезке $[a, b]$:

   - *Создайте отдельную функцию* для вывода таблицы значений функции.

     ```
     void PrintTable(double a, double b, int m);
     ```

     Положите сначала $a = 0.53$, $b = 0.85$, $m = 4$ и выведите “таблицу” из m равномерно распределённых на $[a, b]$ значений. В каждой строчке должен быть номер i, координата x, значения $f(x)$, $g(x)$ и их разность.

```
i = 0  x = 0.530000  f = 0.255564  g = 0.255564  delta = 0.000000e+00
i = 1  x = 0.636667  f = 0.353452  g = 0.353452  delta = -5.551115e-17
i = 2  x = 0.743333  f = 0.457985  g = 0.457985  delta = 5.551115e-17
i = 3  x = 0.850000  f = 0.564422  g = 0.564422  delta = -1.110223e-16
```

   - *Убедитесь, что вы верно организовали цикл,* и строчек в таблице действительно m, а также в ней присутствуют обе границы отрезка. Из-за накапливающейся погрешности нельзя использовать цикл по вещественной переменной, только по целой (номер строчки).

   - *Добавьте чтение с клавиатуры* `a`, `b` и `m`, проверяя диапазон $|x| \leq 1, m > 1$.

   - *Организуйте ровные столбцы*, используя задание ширины поля вывода и точности в форматной строке `printf` (напр., `%8.2lf`).

   - *Оформите таблицу красиво* с помощью знаков “минус”, “плюс” и “вертикальная черта” в качестве границ.

     ```
     +-----+-----+--- ... ---+-------------+
     |  i  |  x  |           |  f(x)-g(x)  |
     +-----+-----+--- ... ---+-------------+
     |  0  | 0.0 |           |  0.00e+00   |
     |  1  | 0.1 |           |  1.57e-16   |
     :     :     :           :             :
     ```

4. (*) **Выведите дополнительную информацию (+1 бонус).** Интересно узнать, сколько слагаемых n потребовалось для достижения необходимой точности, и чему было равно $\varepsilon(s_n) = u_{n+1}$, т.е. машинное “эпсилон”. Добавьте эти два столбца в таблицу.

   Это сделать не так-то просто, мы знаем их только внутри функции `f()`. Есть несколько способов передать их наружу:

   - через глобальные переменные, но их мы сейчас не рассматриваем,
   - через сложный вовзращаемый тип — структуру,
   - через аргументы-указатели.