**Курсовая задача №4** по курсу Фундаментальная информатика

Студент группы: **М8О-103Б-23**, Махмутов Дэниз Ирикович № по списку: **16**, Контакты **lustmodesta@gmail.com** Работа выполнена: « 24» декабря 2023г.

Преподаватель: **Никулин Сергей Петрович**, Входной контроль знаний с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчет сдан « » \_\_\_\_\_\_\_\_\_202 \_\_ г., итоговая оценка \_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Постановка задачи: Составить программу на Си, которая печатает таблицу значений элементарной функции, вычисленной двумя способами: по формуле Тейлора и с помощью встроенных функций языка программирования. В качестве аргементов взять точки разбиения отрезка [0, 1] на n равных частей. Вычисления по формуле Тейлора проводить по экономной в сложностном смысле схеме с точностью e \* k, где e — машинное эпсилон аппаратно-реализованного вещественного типа для данной ЭВМ, а к — экспериментально подбираемый коэффициент, обеспечивающий приемлемую сходимость. Число итераций должно ограничиваться сверху числом порядка 100.**

**Функция, используемая в данной лабораторной работе:**



**Общий метод решения:**

1)Напишем программу для решения поставленной задачи

2)Экспериментально подберем коэффициент к

3)Протестируем программу при различных значениях n

1. **Общие сведения о программе**

ОС:Linux Ubuntu 22.04

Язык программирования: С

Количество строк программы: 62

1. **Функциональное назначение:** Сравнение результатов, полученных функцией и рядом Тейлора, подбор коэффициента к
2. **Описание логической структуры:**

1 выполняется цикл, в котором определяется машинное эпсилон и запрашивается количество разбиений отрезка от пользователя.

2)Вычисляется шаг разбиения и выводится заголовок таблицы с результатами вычислений.

3)Внутри цикла выполняется алгоритм вычисления суммы ряда Тейлора

4)Инициализируются начальные значения коэффициентов и суммы ряда

5)Выполняется цикл, в котором производятся итерации итерационной формулы ряда Тейлора

6)После каждой итерации проверяется точность вычисления и количество итераций для предотвращения бесконечного цикла

7)Результаты суммы ряда, значения функции и номера итерации выводятся и формируют табличку.

8)В конце программы выводится значение машинного эпсилон.

1. **Описание переменных и констант**

|  |  |
| --- | --- |
| **Переменная** | **Назначение** |
| eps | Машинный эпсилон |
| step | Количество разбиений отрезка, вводимое пользователем |
| an, bn | начальные значения коэффициентов ряда |
| bn | Приращение аргумента |
|  | Аргумент функции, итеративно принимающий значение в диапазоне [0;1] |
| steps | количество разбиений интервала |
| x | текущее значение переменной x на интервале |
| sum, sum\_prom | переменные для хранения суммы ряда и промежуточных значений, |
| x | сумма членов ряда Тейлора; присваиваем ей базовое значение — 1 |

1. **Функциональное назначение подпрограмм:**

|  |  |  |
| --- | --- | --- |
| Имя | Вид | Тип возвращаемого значения |
| sn\_1 | double sn\_1(double an, double bn, int n, double x){  double a = an\_1(an, n, x);  double b = bn\_1(bn);  return a \* b;  } | Функция, вычисляющая каждый член ряда Тейлора для данного значения n и x на основе текущих значений an и bn. |
| bn\_1 | double bn\_1(double bn){  return bn + 2;  } | Функция, вычисляющая значение коэффициента bn+1 в формуле для суммы ряда Тейлора. |
| an\_1 | double an\_1(double an, int n, double x){  return (an \* x \* x) / (n + 1);  } | Функция, вычисляющая значение коэффициента an+1 в формуле для суммы ряда Тейлора. |
| func1 | double func1(double x) {  return ((1 + 2 \* (x \* x)) \* exp(x \* x));  } | Функция, для которой вычисляется сумма ряда Тейлора.  ((1 + 2 \* (x \* x)) \* exp(x \* x)) |
| main | int main() {  double A = 0.0, B = 1.0;  double an = 1.0, bn = 1.0, eps = 1.0, x = A, sum, sum\_prom = 0;  int steps;  while (1.0 + eps / 2.0 > 1) {  eps /= 2.0;  }  printf("Введите число разбиения отрезка [0, 1]: ");  scanf("%d", &steps);  double step = (B - A) / steps;  printf("\n==================================================================\n");  printf("|| Шаг x Сумма ряда Тейлора Функция i ||\n");  printf("||--------------------------------------------------------------||\n");  for (int i = 0; i <= steps; i++) {  an = 1.0;  bn = 1.0;  sum = 1.0;  int n = 0;  do {  if (x != 0.0) {  sum\_prom = sn\_1(an, bn, n, x);  sum += sum\_prom;  an = an\_1(an, n, x);  bn = bn\_1(bn);  n++;  }  } while (fabs(sum\_prom) > eps && n < 100);  printf("|| %2d | %.2lf | %.18lf | %.18lf | %2d ||\n", i, x, sum, func1(x), n);  x += step;  }  printf("==================================================================\n");  printf("Машинное эпсилон = %.20f\n", eps);  return 0;  } | «тело» программы |

1. **Входные данные:**

Натуральное число n — количество подотрезков (аргументов функции), заданных в диапозоне от 0 до 1.

Число k – экспериментально подбираемый коэффициент

1. **Выходные данные:**

Таблица формата <значение аргумента> <частичная сумма ряда> <значение функции от текущего аргумента> <количество итераций>

1. **Тестовые примеры:**

|  |  |
| --- | --- |
| **Исходные данные** | **Ожидаемые результаты** |
| **10** | **==================================================================**  **|| Шаг x Сумма ряда Тейлора Функция i ||**  **||--------------------------------------------------------------||**  **|| 0 | 0.00 | 1.000000000000000000 | 1.000000000000000000 | 0 ||**  **|| 1 | 0.10 | 1.030251170425851459 | 1.030251170425851237 | 7 ||**  **|| 2 | 0.20 | 1.124075636127779276 | 1.124075636127779276 | 9 ||**  **|| 3 | 0.30 | 1.291125654772148357 | 1.291125654772148357 | 10 ||**  **|| 4 | 0.40 | 1.549034349709189629 | 1.549034349709189629 | 12 ||**  **|| 5 | 0.50 | 1.926038125031612314 | 1.926038125031612092 | 13 ||**  **|| 6 | 0.60 | 2.465326593043784875 | 2.465326593043784875 | 14 ||**  **|| 7 | 0.70 | 3.231986115511649604 | 3.231986115511650048 | 16 ||**  **|| 8 | 0.80 | 4.323976404815289776 | 4.323976404815288888 | 17 ||**  **|| 9 | 0.90 | 5.889518925092351864 | 5.889518925092352752 | 18 ||**  **|| 10 | 1.00 | 8.154845485377128611 | 8.154845485377132164 | 20 ||**  **==================================================================**  **Машинное эпсилон = 0.00000000000000022204** |
| **100** | **==================================================================**  **|| Шаг x Сумма ряда Тейлора Функция i ||**  **||--------------------------------------------------------------||**  **|| 0 | 0.00 | 1.000000000000000000 | 1.000000000000000000 | 0 ||**  **|| 1 | 0.01 | 1.000300025001166659 | 1.000300025001166659 | 4 ||**  **|| 2 | 0.02 | 1.001200400074676367 | 1.001200400074676367 | 5 ||**  **|| 3 | 0.03 | 1.002702025850745748 | 1.002702025850746193 | 5 ||**  **|| 4 | 0.04 | 1.004806404781124929 | 1.004806404781125151 | 6 ||**  **|| 5 | 0.05 | 1.007515643243824144 | 1.007515643243824144 | 6 ||**  **|| 6 | 0.06 | 1.010832454495041111 | 1.010832454495041333 | 6 ||**  **|| 7 | 0.07 | 1.014760162473605787 | 1.014760162473606009 | 7 ||**  **|| 8 | 0.08 | 1.019302706464798147 | 1.019302706464797703 | 7 ||**  **|| 9 | 0.09 | 1.024464646631953135 | 1.024464646631953357 | 7 ||**  **|| 10 | 0.10 | 1.030251170425851459 | 1.030251170425851237 | 7 ||**  **|| 11 | 0.11 | 1.036668099883457428 | 1.036668099883457428 | 7 ||**  **|| 12 | 0.12 | 1.043721899829232580 | 1.043721899829232358 | 8 ||**  **|| 13 | 0.13 | 1.051419686993860347 | 1.051419686993861013 | 8 ||**  **|| 14 | 0.14 | 1.059769240066933405 | 1.059769240066933182 | 8 ||**  **|| 15 | 0.15 | 1.068779010701845777 | 1.068779010701846000 | 8 ||**  **|| 16 | 0.16 | 1.078458135492929681 | 1.078458135492929681 | 8 ||**  **|| 17 | 0.17 | 1.088816448946634985 | 1.088816448946634985 | 8 ||**  **|| 18 | 0.18 | 1.099864497470449232 | 1.099864497470449010 | 9 ||**  **|| 19 | 0.19 | 1.111613554405126658 | 1.111613554405126658 | 9 ||**  **|| 20 | 0.20 | 1.124075636127779276 | 1.124075636127779276 | 9 ||**  **|| 21 | 0.21 | 1.137263519255402144 | 1.137263519255401922 | 9 ||**  **|| 22 | 0.22 | 1.151190758980488615 | 1.151190758980488837 | 9 ||**  **|| 23 | 0.23 | 1.165871708572574050 | 1.165871708572573828 | 9 ||**  **|| 24 | 0.24 | 1.181321540081754717 | 1.181321540081754717 | 10 ||**  **|| 25 | 0.25 | 1.197556266282591597 | 1.197556266282592041 | 10 ||**  **|| 26 | 0.26 | 1.214592763899167149 | 1.214592763899166927 | 10 ||**  **|| 27 | 0.27 | 1.232448798154602621 | 1.232448798154602621 | 10 ||**  **|| 28 | 0.28 | 1.251143048690928961 | 1.251143048690928961 | 10 ||**  **|| 29 | 0.29 | 1.270695136907898037 | 1.270695136907897815 | 10 ||**  **|| 30 | 0.30 | 1.291125654772148579 | 1.291125654772148357 | 10 ||**  **|| 31 | 0.31 | 1.312456195151074922 | 1.312456195151074700 | 11 ||**  **|| 32 | 0.32 | 1.334709383728790666 | 1.334709383728790666 | 11 ||**  **|| 33 | 0.33 | 1.357908912564788073 | 1.357908912564788073 | 11 ||**  **|| 34 | 0.34 | 1.382079575359210111 | 1.382079575359210333 | 11 ||**  **|| 35 | 0.35 | 1.407247304492144080 | 1.407247304492144080 | 11 ||**  **|| 36 | 0.36 | 1.433439209907972467 | 1.433439209907972911 | 11 ||**  **|| 37 | 0.37 | 1.460683619919630027 | 1.460683619919630249 | 11 ||**  **|| 38 | 0.38 | 1.489010124011585390 | 1.489010124011584946 | 12 ||**  **|| 39 | 0.39 | 1.518449617724539014 | 1.518449617724539014 | 12 ||**  **|| 40 | 0.40 | 1.549034349709190073 | 1.549034349709190295 | 12 ||**  **|| 41 | 0.41 | 1.580797971040983185 | 1.580797971040982963 | 12 ||**  **|| 42 | 0.42 | 1.613775586892548297 | 1.613775586892548075 | 12 ||**  **|| 43 | 0.43 | 1.648003810665557234 | 1.648003810665557456 | 12 ||**  **|| 44 | 0.44 | 1.683520820688986674 | 1.683520820688986674 | 12 ||**  **|| 45 | 0.45 | 1.720366419596291108 | 1.720366419596291330 | 13 ||**  **|| 46 | 0.46 | 1.758582096499799574 | 1.758582096499799352 | 13 ||**  **|| 47 | 0.47 | 1.798211092086706042 | 1.798211092086705820 | 13 ||**  **|| 48 | 0.48 | 1.839298466767429030 | 1.839298466767429252 | 13 ||**  **|| 49 | 0.49 | 1.881891172013803626 | 1.881891172013803404 | 13 ||**  **|| 50 | 0.50 | 1.926038125031613202 | 1.926038125031613424 | 13 ||**  **|| 51 | 0.51 | 1.971790286919375967 | 1.971790286919376189 | 13 ||**  **|| 52 | 0.52 | 2.019200744473053533 | 2.019200744473054421 | 13 ||**  **|| 53 | 0.53 | 2.068324795804536986 | 2.068324795804536986 | 14 ||**  **|| 54 | 0.54 | 2.119220039950340251 | 2.119220039950340695 | 14 ||**  **|| 55 | 0.55 | 2.171946470655982786 | 2.171946470655982786 | 14 ||**  **|| 56 | 0.56 | 2.226566574531000597 | 2.226566574531000597 | 14 ||**  **|| 57 | 0.57 | 2.283145433779559141 | 2.283145433779559585 | 14 ||**  **|| 58 | 0.58 | 2.341750833722124447 | 2.341750833722124892 | 14 ||**  **|| 59 | 0.59 | 2.402453375334697494 | 2.402453375334696606 | 14 ||**  **|| 60 | 0.60 | 2.465326593043787096 | 2.465326593043787096 | 14 ||**  **|| 61 | 0.61 | 2.530447078027519225 | 2.530447078027518337 | 15 ||**  **|| 62 | 0.62 | 2.597894607286158397 | 2.597894607286159285 | 15 ||**  **|| 63 | 0.63 | 2.667752278758954709 | 2.667752278758954265 | 15 ||**  **|| 64 | 0.64 | 2.740106652778430796 | 2.740106652778431240 | 15 ||**  **|| 65 | 0.65 | 2.815047900168388573 | 2.815047900168389461 | 15 ||**  **|| 66 | 0.66 | 2.892669957307660411 | 2.892669957307659967 | 15 ||**  **|| 67 | 0.67 | 2.973070688498422598 | 2.973070688498423042 | 15 ||**  **|| 68 | 0.68 | 3.056352055995461559 | 3.056352055995462003 | 16 ||**  **|| 69 | 0.69 | 3.142620298071293394 | 3.142620298071292506 | 16 ||**  **|| 70 | 0.70 | 3.231986115511653157 | 3.231986115511653601 | 16 ||**  **|| 71 | 0.71 | 3.324564866956427167 | 3.324564866956427167 | 16 ||**  **|| 72 | 0.72 | 3.420476773522809211 | 3.420476773522809655 | 16 ||**  **|| 73 | 0.73 | 3.519847133170430631 | 3.519847133170430187 | 16 ||**  **|| 74 | 0.74 | 3.622806545292261315 | 3.622806545292262204 | 16 ||**  **|| 75 | 0.75 | 3.729491146040639027 | 3.729491146040639471 | 16 ||**  **|| 76 | 0.76 | 3.840042854924551996 | 3.840042854924552884 | 17 ||**  **|| 77 | 0.77 | 3.954609633242719635 | 3.954609633242718747 | 17 ||**  **|| 78 | 0.78 | 4.073345754946808839 | 4.073345754946808839 | 17 ||**  **|| 79 | 0.79 | 4.196412090560748709 | 4.196412090560748709 | 17 ||**  **|| 80 | 0.80 | 4.323976404815295993 | 4.323976404815295105 | 17 ||**  **|| 81 | 0.81 | 4.456213668692188712 | 4.456213668692188712 | 17 ||**  **|| 82 | 0.82 | 4.593306386609257608 | 4.593306386609257608 | 17 ||**  **|| 83 | 0.83 | 4.735444939516983531 | 4.735444939516982643 | 17 ||**  **|| 84 | 0.84 | 4.882827944718329150 | 4.882827944718329150 | 18 ||**  **|| 85 | 0.85 | 5.035662633267265598 | 5.035662633267265598 | 18 ||**  **|| 86 | 0.86 | 5.194165245847467638 | 5.194165245847468526 | 18 ||**  **|| 87 | 0.87 | 5.358561448081317735 | 5.358561448081317735 | 18 ||**  **|| 88 | 0.88 | 5.529086766270708608 | 5.529086766270710385 | 18 ||**  **|| 89 | 0.89 | 5.705987044625450366 | 5.705987044625451254 | 18 ||**  **|| 90 | 0.90 | 5.889518925092366075 | 5.889518925092366075 | 18 ||**  **|| 91 | 0.91 | 6.079950350958829297 | 6.079950350958829297 | 18 ||**  **|| 92 | 0.92 | 6.277561095468456642 | 6.277561095468453978 | 19 ||**  **|| 93 | 0.93 | 6.482643316754359120 | 6.482643316754360008 | 19 ||**  **|| 94 | 0.94 | 6.695502140466927266 | 6.695502140466929042 | 19 ||**  **|| 95 | 0.95 | 6.916456271548634405 | 6.916456271548633516 | 19 ||**  **|| 96 | 0.96 | 7.145838636688417012 | 7.145838636688417900 | 19 ||**  **|| 97 | 0.97 | 7.383997059072620495 | 7.383997059072621383 | 19 ||**  **|| 98 | 0.98 | 7.631294967138908092 | 7.631294967138908092 | 19 ||**  **|| 99 | 0.99 | 7.888112139134145551 | 7.888112139134145551 | 19 ||**  **|| 100 | 1.00 | 8.154845485377148151 | 8.154845485377153480 | 20 ||**  **==================================================================**  **Машинное эпсилон = 0.00000000000000022204** |
| **40** | **==================================================================**  **|| Шаг x Сумма ряда Тейлора Функция i ||**  **||--------------------------------------------------------------||**  **|| 0 | 0.00 | 1.000000000000000000 | 1.000000000000000000 | 0 ||**  **|| 1 | 0.03 | 1.001875976847388161 | 1.001875976847387939 | 5 ||**  **|| 2 | 0.05 | 1.007515643243824144 | 1.007515643243824144 | 6 ||**  **|| 3 | 0.08 | 1.016954309580041738 | 1.016954309580041738 | 7 ||**  **|| 4 | 0.10 | 1.030251170425851459 | 1.030251170425851237 | 7 ||**  **|| 5 | 0.12 | 1.047489824480019704 | 1.047489824480019704 | 8 ||**  **|| 6 | 0.15 | 1.068779010701845777 | 1.068779010701846000 | 8 ||**  **|| 7 | 0.17 | 1.094253568961889078 | 1.094253568961889078 | 9 ||**  **|| 8 | 0.20 | 1.124075636127779054 | 1.124075636127779276 | 9 ||**  **|| 9 | 0.22 | 1.158436091231196619 | 1.158436091231196396 | 9 ||**  **|| 10 | 0.25 | 1.197556266282591597 | 1.197556266282591819 | 10 ||**  **|| 11 | 0.27 | 1.241689942452914641 | 1.241689942452914641 | 10 ||**  **|| 12 | 0.30 | 1.291125654772148357 | 1.291125654772148135 | 10 ||**  **|| 13 | 0.33 | 1.346189332253170168 | 1.346189332253170168 | 11 ||**  **|| 14 | 0.35 | 1.407247304492143858 | 1.407247304492143858 | 11 ||**  **|| 15 | 0.38 | 1.474709710385569972 | 1.474709710385569750 | 12 ||**  **|| 16 | 0.40 | 1.549034349709189851 | 1.549034349709189629 | 12 ||**  **|| 17 | 0.43 | 1.630731024003757801 | 1.630731024003757801 | 12 ||**  **|| 18 | 0.45 | 1.720366419596290886 | 1.720366419596290886 | 13 ||**  **|| 19 | 0.48 | 1.818569592753527253 | 1.818569592753527475 | 13 ||**  **|| 20 | 0.50 | 1.926038125031612980 | 1.926038125031612758 | 13 ||**  **|| 21 | 0.53 | 2.043545025982889562 | 2.043545025982889118 | 14 ||**  **|| 22 | 0.55 | 2.171946470655981898 | 2.171946470655982342 | 14 ||**  **|| 23 | 0.58 | 2.312190470949167409 | 2.312190470949166521 | 14 ||**  **|| 24 | 0.60 | 2.465326593043786652 | 2.465326593043786652 | 14 ||**  **|| 25 | 0.63 | 2.632516848077159910 | 2.632516848077160354 | 15 ||**  **|| 26 | 0.65 | 2.815047900168387685 | 2.815047900168388573 | 15 ||**  **|| 27 | 0.68 | 3.014344755179129365 | 3.014344755179129809 | 15 ||**  **|| 28 | 0.70 | 3.231986115511652269 | 3.231986115511653157 | 16 ||**  **|| 29 | 0.73 | 3.469721611205776490 | 3.469721611205776934 | 16 ||**  **|| 30 | 0.75 | 3.729491146040638583 | 3.729491146040638139 | 16 ||**  **|| 31 | 0.78 | 4.013446629790219156 | 4.013446629790217379 | 17 ||**  **|| 32 | 0.80 | 4.323976404815294217 | 4.323976404815293328 | 17 ||**  **|| 33 | 0.83 | 4.663732717482020007 | 4.663732717482018231 | 17 ||**  **|| 34 | 0.85 | 5.035662633267264710 | 5.035662633267262933 | 18 ||**  **|| 35 | 0.88 | 5.443042849754267110 | 5.443042849754267110 | 18 ||**  **|| 36 | 0.90 | 5.889518925092362522 | 5.889518925092363411 | 18 ||**  **|| 37 | 0.93 | 6.379149512110671161 | 6.379149512110670273 | 19 ||**  **|| 38 | 0.95 | 6.916456271548630852 | 6.916456271548631740 | 19 ||**  **|| 39 | 0.98 | 7.506480233431978455 | 7.506480233431978455 | 19 ||**  **|| 40 | 1.00 | 8.154845485377142822 | 8.154845485377148151 | 20 ||**  **==================================================================**  **Машинное эпсилон = 0.00000000000000022204** |

1. **Решение**

#include <stdio.h>

#include <math.h>

double func1(double x) {

    return ((1 + 2 \* (x \* x)) \* exp(x \* x));

}

double an\_1(double an, int n, double x){

    return (an \* x \* x) / (n + 1);

}

double bn\_1(double bn){

    return bn + 2;

}

double sn\_1(double an, double bn, int n, double x){

    double a = an\_1(an, n, x);

    double b = bn\_1(bn);

    return a \* b;

}

int main() {

    double A = 0.0, B = 1.0;

    double an = 1.0, bn = 1.0, eps = 1.0, x = A, sum, sum\_prom = 0;

    int steps;

    while (1.0 + eps / 2.0 > 1) {

        eps /= 2.0;

    }

    printf("Введите число разбиения отрезка [0, 1]: ");

    scanf("%d", &steps);

    double step = (B - A) / steps;

    printf("\n==================================================================\n");

    printf("|| Шаг   x     Сумма ряда Тейлора      Функция               i  ||\n");

    printf("||--------------------------------------------------------------||\n");

    for (int i = 0; i <= steps; i++) {

        an = 1.0;

        bn = 1.0;

        sum = 1.0;

        int n = 0;

        do {

            if (x != 0.0) {

                sum\_prom = sn\_1(an, bn, n, x);

                sum += sum\_prom;

                an = an\_1(an, n, x);

                bn = bn\_1(bn);

                n++;

            }

        } while (fabs(sum\_prom) > eps && n < 100);

        printf("|| %2d | %.2lf | %.18lf | %.18lf | %2d ||\n", i, x, sum, func1(x), n);

        x += step;

    }

    printf("==================================================================\n");

    printf("Машинное эпсилон = %.20f\n", eps);

    return 0;

}

**Распечатка протокола**  (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем).

#include <stdio.h>

#include <math.h>

double func1(double x) {

return ((1 + 2 \* (x \* x)) \* exp(x \* x));

}

double an\_1(double an, int n, double x){

return (an \* x \* x) / (n + 1);

}

double bn\_1(double bn){

return bn + 2;

}

double sn\_1(double an, double bn, int n, double x){

double a = an\_1(an, n, x);

double b = bn\_1(bn);

return a \* b;

}

int main() {

double A = 0.0, B = 1.0;

double an = 1.0, bn = 1.0, eps = 1.0, x = A, sum, sum\_prom = 0;

int steps;

while (1.0 + eps / 2.0 > 1) {

eps /= 2.0;

}

printf("Введите число разбиения отрезка [0, 1]: ");

scanf("%d", &steps);

double step = (B - A) / steps;

printf("\n==================================================================\n");

printf("|| Шаг x Сумма ряда Тейлора Функция i ||\n");

printf("||--------------------------------------------------------------||\n");

for (int i = 0; i <= steps; i++) {

an = 1.0;

bn = 1.0;

sum = 1.0;

int n = 0;

do {

if (x != 0.0) {

sum\_prom = sn\_1(an, bn, n, x);

sum += sum\_prom;

an = an\_1(an, n, x);

bn = bn\_1(bn);

n++;

}

} while (fabs(sum\_prom) > eps && n < 100);

printf("|| %2d | %.2lf | %.18lf | %.18lf | %2d ||\n", i, x, sum, func1(x), n);

x += step;

}

printf("==================================================================\n");

printf("Машинное эпсилон = %.20f\n", eps);

return 0;

}

unix@DESKTOP-MPQDBS2:~/labs$ gcc c.c -lm

unix@DESKTOP-MPQDBS2:~/labs$ ./a.out

Введите число разбиения отрезка [0, 1]: 10

==================================================================

|| Шаг x Сумма ряда Тейлора Функция i ||

||--------------------------------------------------------------||

|| 0 | 0.00 | 1.000000000000000000 | 1.000000000000000000 | 0 ||

|| 1 | 0.10 | 1.030251170425851459 | 1.030251170425851237 | 7 ||

|| 2 | 0.20 | 1.124075636127779276 | 1.124075636127779276 | 9 ||

|| 3 | 0.30 | 1.291125654772148357 | 1.291125654772148357 | 10 ||

|| 4 | 0.40 | 1.549034349709189629 | 1.549034349709189629 | 12 ||

|| 5 | 0.50 | 1.926038125031612314 | 1.926038125031612092 | 13 ||

|| 6 | 0.60 | 2.465326593043784875 | 2.465326593043784875 | 14 ||

|| 7 | 0.70 | 3.231986115511649604 | 3.231986115511650048 | 16 ||

|| 8 | 0.80 | 4.323976404815289776 | 4.323976404815288888 | 17 ||

|| 9 | 0.90 | 5.889518925092351864 | 5.889518925092352752 | 18 ||

|| 10 | 1.00 | 8.154845485377128611 | 8.154845485377132164 | 20 ||

==================================================================

Машинное эпсилон = 0.00000000000000022204

unix@DESKTOP-MPQDBS2:~/labs$ ./a.out

Введите число разбиения отрезка [0, 1]: 100

==================================================================

|| Шаг x Сумма ряда Тейлора Функция i ||

||--------------------------------------------------------------||

|| 0 | 0.00 | 1.000000000000000000 | 1.000000000000000000 | 0 ||

|| 1 | 0.01 | 1.000300025001166659 | 1.000300025001166659 | 4 ||

|| 2 | 0.02 | 1.001200400074676367 | 1.001200400074676367 | 5 ||

|| 3 | 0.03 | 1.002702025850745748 | 1.002702025850746193 | 5 ||

|| 4 | 0.04 | 1.004806404781124929 | 1.004806404781125151 | 6 ||

|| 5 | 0.05 | 1.007515643243824144 | 1.007515643243824144 | 6 ||

|| 6 | 0.06 | 1.010832454495041111 | 1.010832454495041333 | 6 ||

|| 7 | 0.07 | 1.014760162473605787 | 1.014760162473606009 | 7 ||

|| 8 | 0.08 | 1.019302706464798147 | 1.019302706464797703 | 7 ||

|| 9 | 0.09 | 1.024464646631953135 | 1.024464646631953357 | 7 ||

|| 10 | 0.10 | 1.030251170425851459 | 1.030251170425851237 | 7 ||

|| 11 | 0.11 | 1.036668099883457428 | 1.036668099883457428 | 7 ||

|| 12 | 0.12 | 1.043721899829232580 | 1.043721899829232358 | 8 ||

|| 13 | 0.13 | 1.051419686993860347 | 1.051419686993861013 | 8 ||

|| 14 | 0.14 | 1.059769240066933405 | 1.059769240066933182 | 8 ||

|| 15 | 0.15 | 1.068779010701845777 | 1.068779010701846000 | 8 ||

|| 16 | 0.16 | 1.078458135492929681 | 1.078458135492929681 | 8 ||

|| 17 | 0.17 | 1.088816448946634985 | 1.088816448946634985 | 8 ||

|| 18 | 0.18 | 1.099864497470449232 | 1.099864497470449010 | 9 ||

|| 19 | 0.19 | 1.111613554405126658 | 1.111613554405126658 | 9 ||

|| 20 | 0.20 | 1.124075636127779276 | 1.124075636127779276 | 9 ||

|| 21 | 0.21 | 1.137263519255402144 | 1.137263519255401922 | 9 ||

|| 22 | 0.22 | 1.151190758980488615 | 1.151190758980488837 | 9 ||

|| 23 | 0.23 | 1.165871708572574050 | 1.165871708572573828 | 9 ||

|| 24 | 0.24 | 1.181321540081754717 | 1.181321540081754717 | 10 ||

|| 25 | 0.25 | 1.197556266282591597 | 1.197556266282592041 | 10 ||

|| 26 | 0.26 | 1.214592763899167149 | 1.214592763899166927 | 10 ||

|| 27 | 0.27 | 1.232448798154602621 | 1.232448798154602621 | 10 ||

|| 28 | 0.28 | 1.251143048690928961 | 1.251143048690928961 | 10 ||

|| 29 | 0.29 | 1.270695136907898037 | 1.270695136907897815 | 10 ||

|| 30 | 0.30 | 1.291125654772148579 | 1.291125654772148357 | 10 ||

|| 31 | 0.31 | 1.312456195151074922 | 1.312456195151074700 | 11 ||

|| 32 | 0.32 | 1.334709383728790666 | 1.334709383728790666 | 11 ||

|| 33 | 0.33 | 1.357908912564788073 | 1.357908912564788073 | 11 ||

|| 34 | 0.34 | 1.382079575359210111 | 1.382079575359210333 | 11 ||

|| 35 | 0.35 | 1.407247304492144080 | 1.407247304492144080 | 11 ||

|| 36 | 0.36 | 1.433439209907972467 | 1.433439209907972911 | 11 ||

|| 37 | 0.37 | 1.460683619919630027 | 1.460683619919630249 | 11 ||

|| 38 | 0.38 | 1.489010124011585390 | 1.489010124011584946 | 12 ||

|| 39 | 0.39 | 1.518449617724539014 | 1.518449617724539014 | 12 ||

|| 40 | 0.40 | 1.549034349709190073 | 1.549034349709190295 | 12 ||

|| 41 | 0.41 | 1.580797971040983185 | 1.580797971040982963 | 12 ||

|| 42 | 0.42 | 1.613775586892548297 | 1.613775586892548075 | 12 ||

|| 43 | 0.43 | 1.648003810665557234 | 1.648003810665557456 | 12 ||

|| 44 | 0.44 | 1.683520820688986674 | 1.683520820688986674 | 12 ||

|| 45 | 0.45 | 1.720366419596291108 | 1.720366419596291330 | 13 ||

|| 46 | 0.46 | 1.758582096499799574 | 1.758582096499799352 | 13 ||

|| 47 | 0.47 | 1.798211092086706042 | 1.798211092086705820 | 13 ||

|| 48 | 0.48 | 1.839298466767429030 | 1.839298466767429252 | 13 ||

|| 49 | 0.49 | 1.881891172013803626 | 1.881891172013803404 | 13 ||

|| 50 | 0.50 | 1.926038125031613202 | 1.926038125031613424 | 13 ||

|| 51 | 0.51 | 1.971790286919375967 | 1.971790286919376189 | 13 ||

|| 52 | 0.52 | 2.019200744473053533 | 2.019200744473054421 | 13 ||

|| 53 | 0.53 | 2.068324795804536986 | 2.068324795804536986 | 14 ||

|| 54 | 0.54 | 2.119220039950340251 | 2.119220039950340695 | 14 ||

|| 55 | 0.55 | 2.171946470655982786 | 2.171946470655982786 | 14 ||

|| 56 | 0.56 | 2.226566574531000597 | 2.226566574531000597 | 14 ||

|| 57 | 0.57 | 2.283145433779559141 | 2.283145433779559585 | 14 ||

|| 58 | 0.58 | 2.341750833722124447 | 2.341750833722124892 | 14 ||

|| 59 | 0.59 | 2.402453375334697494 | 2.402453375334696606 | 14 ||

|| 60 | 0.60 | 2.465326593043787096 | 2.465326593043787096 | 14 ||

|| 61 | 0.61 | 2.530447078027519225 | 2.530447078027518337 | 15 ||

|| 62 | 0.62 | 2.597894607286158397 | 2.597894607286159285 | 15 ||

|| 63 | 0.63 | 2.667752278758954709 | 2.667752278758954265 | 15 ||

|| 64 | 0.64 | 2.740106652778430796 | 2.740106652778431240 | 15 ||

|| 65 | 0.65 | 2.815047900168388573 | 2.815047900168389461 | 15 ||

|| 66 | 0.66 | 2.892669957307660411 | 2.892669957307659967 | 15 ||

|| 67 | 0.67 | 2.973070688498422598 | 2.973070688498423042 | 15 ||

|| 68 | 0.68 | 3.056352055995461559 | 3.056352055995462003 | 16 ||

|| 69 | 0.69 | 3.142620298071293394 | 3.142620298071292506 | 16 ||

|| 70 | 0.70 | 3.231986115511653157 | 3.231986115511653601 | 16 ||

|| 71 | 0.71 | 3.324564866956427167 | 3.324564866956427167 | 16 ||

|| 72 | 0.72 | 3.420476773522809211 | 3.420476773522809655 | 16 ||

|| 73 | 0.73 | 3.519847133170430631 | 3.519847133170430187 | 16 ||

|| 74 | 0.74 | 3.622806545292261315 | 3.622806545292262204 | 16 ||

|| 75 | 0.75 | 3.729491146040639027 | 3.729491146040639471 | 16 ||

|| 76 | 0.76 | 3.840042854924551996 | 3.840042854924552884 | 17 ||

|| 77 | 0.77 | 3.954609633242719635 | 3.954609633242718747 | 17 ||

|| 78 | 0.78 | 4.073345754946808839 | 4.073345754946808839 | 17 ||

|| 79 | 0.79 | 4.196412090560748709 | 4.196412090560748709 | 17 ||

|| 80 | 0.80 | 4.323976404815295993 | 4.323976404815295105 | 17 ||

|| 81 | 0.81 | 4.456213668692188712 | 4.456213668692188712 | 17 ||

|| 82 | 0.82 | 4.593306386609257608 | 4.593306386609257608 | 17 ||

|| 83 | 0.83 | 4.735444939516983531 | 4.735444939516982643 | 17 ||

|| 84 | 0.84 | 4.882827944718329150 | 4.882827944718329150 | 18 ||

|| 85 | 0.85 | 5.035662633267265598 | 5.035662633267265598 | 18 ||

|| 86 | 0.86 | 5.194165245847467638 | 5.194165245847468526 | 18 ||

|| 87 | 0.87 | 5.358561448081317735 | 5.358561448081317735 | 18 ||

|| 88 | 0.88 | 5.529086766270708608 | 5.529086766270710385 | 18 ||

|| 89 | 0.89 | 5.705987044625450366 | 5.705987044625451254 | 18 ||

|| 90 | 0.90 | 5.889518925092366075 | 5.889518925092366075 | 18 ||

|| 91 | 0.91 | 6.079950350958829297 | 6.079950350958829297 | 18 ||

|| 92 | 0.92 | 6.277561095468456642 | 6.277561095468453978 | 19 ||

|| 93 | 0.93 | 6.482643316754359120 | 6.482643316754360008 | 19 ||

|| 94 | 0.94 | 6.695502140466927266 | 6.695502140466929042 | 19 ||

|| 95 | 0.95 | 6.916456271548634405 | 6.916456271548633516 | 19 ||

|| 96 | 0.96 | 7.145838636688417012 | 7.145838636688417900 | 19 ||

|| 97 | 0.97 | 7.383997059072620495 | 7.383997059072621383 | 19 ||

|| 98 | 0.98 | 7.631294967138908092 | 7.631294967138908092 | 19 ||

|| 99 | 0.99 | 7.888112139134145551 | 7.888112139134145551 | 19 ||

|| 100 | 1.00 | 8.154845485377148151 | 8.154845485377153480 | 20 ||

==================================================================

Машинное эпсилон = 0.00000000000000022204

uunix@DESKTOP-MPQDBS2:~/labs$ ./a.out

Введите число разбиения отрезка [0, 1]: 40

==================================================================

|| Шаг x Сумма ряда Тейлора Функция i ||

||--------------------------------------------------------------||

|| 0 | 0.00 | 1.000000000000000000 | 1.000000000000000000 | 0 ||

|| 1 | 0.03 | 1.001875976847388161 | 1.001875976847387939 | 5 ||

|| 2 | 0.05 | 1.007515643243824144 | 1.007515643243824144 | 6 ||

|| 3 | 0.08 | 1.016954309580041738 | 1.016954309580041738 | 7 ||

|| 4 | 0.10 | 1.030251170425851459 | 1.030251170425851237 | 7 ||

|| 5 | 0.12 | 1.047489824480019704 | 1.047489824480019704 | 8 ||

|| 6 | 0.15 | 1.068779010701845777 | 1.068779010701846000 | 8 ||

|| 7 | 0.17 | 1.094253568961889078 | 1.094253568961889078 | 9 ||

|| 8 | 0.20 | 1.124075636127779054 | 1.124075636127779276 | 9 ||

|| 9 | 0.22 | 1.158436091231196619 | 1.158436091231196396 | 9 ||

|| 10 | 0.25 | 1.197556266282591597 | 1.197556266282591819 | 10 ||

|| 11 | 0.27 | 1.241689942452914641 | 1.241689942452914641 | 10 ||

|| 12 | 0.30 | 1.291125654772148357 | 1.291125654772148135 | 10 ||

|| 13 | 0.33 | 1.346189332253170168 | 1.346189332253170168 | 11 ||

|| 14 | 0.35 | 1.407247304492143858 | 1.407247304492143858 | 11 ||

|| 15 | 0.38 | 1.474709710385569972 | 1.474709710385569750 | 12 ||

|| 16 | 0.40 | 1.549034349709189851 | 1.549034349709189629 | 12 ||

|| 17 | 0.43 | 1.630731024003757801 | 1.630731024003757801 | 12 ||

|| 18 | 0.45 | 1.720366419596290886 | 1.720366419596290886 | 13 ||

|| 19 | 0.48 | 1.818569592753527253 | 1.818569592753527475 | 13 ||

|| 20 | 0.50 | 1.926038125031612980 | 1.926038125031612758 | 13 ||

|| 21 | 0.53 | 2.043545025982889562 | 2.043545025982889118 | 14 ||

|| 22 | 0.55 | 2.171946470655981898 | 2.171946470655982342 | 14 ||

|| 23 | 0.58 | 2.312190470949167409 | 2.312190470949166521 | 14 ||

|| 24 | 0.60 | 2.465326593043786652 | 2.465326593043786652 | 14 ||

|| 25 | 0.63 | 2.632516848077159910 | 2.632516848077160354 | 15 ||

|| 26 | 0.65 | 2.815047900168387685 | 2.815047900168388573 | 15 ||

|| 27 | 0.68 | 3.014344755179129365 | 3.014344755179129809 | 15 ||

|| 28 | 0.70 | 3.231986115511652269 | 3.231986115511653157 | 16 ||

|| 29 | 0.73 | 3.469721611205776490 | 3.469721611205776934 | 16 ||

|| 30 | 0.75 | 3.729491146040638583 | 3.729491146040638139 | 16 ||

|| 31 | 0.78 | 4.013446629790219156 | 4.013446629790217379 | 17 ||

|| 32 | 0.80 | 4.323976404815294217 | 4.323976404815293328 | 17 ||

|| 33 | 0.83 | 4.663732717482020007 | 4.663732717482018231 | 17 ||

|| 34 | 0.85 | 5.035662633267264710 | 5.035662633267262933 | 18 ||

|| 35 | 0.88 | 5.443042849754267110 | 5.443042849754267110 | 18 ||

|| 36 | 0.90 | 5.889518925092362522 | 5.889518925092363411 | 18 ||

|| 37 | 0.93 | 6.379149512110671161 | 6.379149512110670273 | 19 ||

|| 38 | 0.95 | 6.916456271548630852 | 6.916456271548631740 | 19 ||

|| 39 | 0.98 | 7.506480233431978455 | 7.506480233431978455 | 19 ||

|| 40 | 1.00 | 8.154845485377142822 | 8.154845485377148151 | 20 ||

==================================================================

Машинное эпсилон = 0.00000000000000022204

**Дневник отладки** должен содержать дату и время сеансов отладки и основные события (ошибки в сценарии и программе, нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб. или дом. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  |  |  |  |  |  |  |

1. **Замечания автора** по существу работы: Отсутствуют
2. **Вывод:** В ходе работы я научился реализовывать ряды Тейлора. Научился с помощью алгоритма определять машинное эпсилон, работать с машинной точностью в языке Си.

Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_