# Индивидуальное задание 2 — RISC-V (FPU). Гиперболический синус через степенной ряд

**Студент:** Бюрчиев Тимур Зольванович
**Группа:** 248
**Вариант:** 2

**Условие задачи:**
Разработать программу, вычисляющую с помощью степенного ряда с точностью не хуже 0,1% значение функции гиперболического синуса

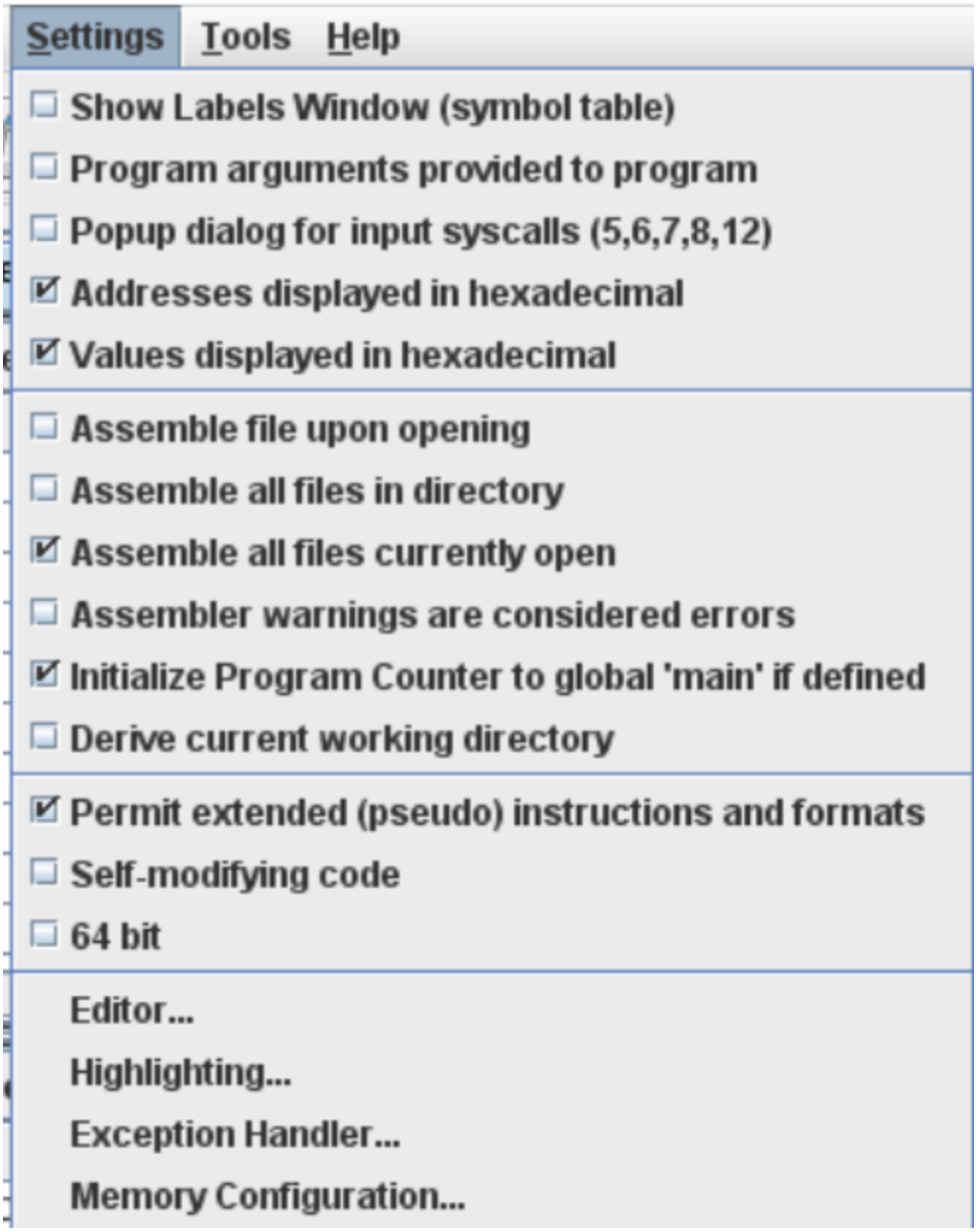
$$sh(x) = \frac{e^x - e^{-x}}{2}$$

для заданного параметра *x*.

---

## Запуск и состав проекта

### Настройки RARS




**Самое важное:**

- Assemble all files currently open
- Initialize Program Counter to global 'main' if defined

> Не держать открытыми **одновременно** `main_autotests.asm`, `main.asm`, так как в обоих прописан main. Сначала открыть 3 файла: `sinh_series.asm`, `macros.asm`, `main.asm`. Потом заменить `main.asm` на `main_autotests.asm`.

**Состав файлов:**

- `sinh_series.asm` — подпрограмма, которая вычисляет `sinh(x)` по ряду.
- `main_autotests.asm` — автономные тесты: прогон набора `x`, печать `x/calc/ref/rel/status` блоками и сводки. Использует макросы.
- `macros.asm` — библиотека макросов.

- `main.asm` — интерактивный ввод `x` с клавиатуры и вывод результата.

---

# Описание метода

Используется разложение в степенной ряд (ряд Тейлора):

$$\sinh(x) = \sum_{k=0}^{\infty} \frac{x^{2k+1}}{(2k+1)!}$$

Член ряда обновляется рекуррентно без факториалов:

$$\text{term}_{k+1} = \text{term}_k \cdot \frac{x^2}{(2k+2)(2k+3)}$$

Сумма накапливается в `sum`. Остановка по **относительной точности**:

$$|\text{term}| \le \varepsilon_{\text{rel}} \cdot \max(1,\ |\text{sum}|)$$

где `sum` — частичная сумма, `eps_rel = 1e-3`. Также предусмотрен **жёсткий лимит итераций** `max_iter` для защиты от зацикливания.

**Выбор метрики сравнения:**
Для тестов из RARS применяется относительная ошибка

$$\text{rel} = \frac{|calc - ref|}{\max(1,\ |ref|)}$$

Порог прохождения: `thr_rel = 1e-3` (0.1%).

---

# Набор тестов и покрытие

**Входной набор** `XS`:
`0.0, ±0.001, ±0.5, ±1.0, ±2.0, ±5.0, ±10.0, ±15.0, ±20.0` (всего 17 значений)

**Покрывает:**

- Ноль и окрестность нуля
- Средние |x| (сходимость ряда и точность).
- Большие |x| до 20 (потенциальный рост числа членов/переполнения).
- Чётную/нечётную симметрию (знаки ±).

**Метрика сравнения:**

$$\text{rel} = \frac{|calc - ref|}{\max(1, |ref|)}, \quad \text{threshold} = 10^{-3}$$

где `ref = math.sinh(x)` (эталон в Python).

---

# Результаты тестов

## С клавиатуры

```
Assemble: assembling C:\Users\timur\Riscv Projects\ihw-2\ihw 9\main.asm,
C:\Users\timur\Riscv Projects\ihw-2\ihw 9\macros.asm, C:\Users\timur\Riscv Projects\ihw-2\ihw 9\sinh_series.asm

Assemble: operation completed successfully.
```

```
Note: |x| > 20 may slow convergence or overflow.
Enter x (double): 12
Using eps_rel = 1e-3, max_iter = 200
sinh(x) = 81362.14270300337
```

## Авто-тесты

Формат вывода по каждому тесту:

```
-----
Test #<i>
x=<значение>
calc=<значение из sinh_series>
ref=<значение math.sinh>
rel=<относительная ошибка>
status: OK|FAIL
```

```
Assemble: assembling C:\Users\timur\Riscv Projects\ihw-2\ihw 9\main_autotests.asm,
C:\Users\timur\Riscv Projects\ihw-2\ihw 9\macros.asm, C:\Users\timur\Riscv Projects\ihw-2\ihw 9\sinh_series.asm

Assemble: operation completed successfully.
```

```
Test #14
x=15.0
calc=1634067.477332553
ref=1634508.6862359024
rel=2.6993365472136277E-4
status: OK
-----
Test #15
x=-15.0
calc=-1634067.477332553
ref=-1634508.6862359024
rel=2.6993365472136277E-4
status: OK
-----
Test #16
x=20.0
calc=2.4254725709287572E8
ref=2.4258259770489514E8
rel=1.4568486096604497E-4
status: OK
-----
Test #17
x=-20.0
calc=-2.4254725709287572E8
ref=-2.4258259770489514E8
rel=1.4568486096604497E-4
status: OK
-----
Passed: 17 / 17
```

## Full output

```
-----
Test #1
x=0.0
```

```
calc=0.0
ref=0.0
rel=0.0
status: OK
-----
Test #2
x=0.001
calc=0.0010000001666666666
ref=0.001000000166666675
rel=8.456776945386935E-18
status: OK
-----
Test #3
x=-0.001
calc=-0.0010000001666666666
ref=-0.001000000166666675
rel=8.456776945386935E-18
status: OK
-----
Test #4
x=0.5
calc=0.52109375
ref=0.52109530549374 74
rel=1.5554937473627461E-6
status: OK
-----
Test #5
x=-0.5
calc=-0.52109375
ref=-0.52109530549374 74
rel=1.5554937473627461E-6
status: OK
-----
Test #6
x=1.0
calc=1.1751984126984127
ref=1.1752011936438014
rel=2.3663568448341467E-6
status: OK
-----
Test #7
x=-1.0
calc=-1.1751984126984127
ref=-1.1752011936438014
rel=2.3663568448341467E-6
status: OK
```

```
-----
Test #8
x=2.0
calc=3.626807760141093
ref=3.626860407847019
rel=1.4516055211840474E-5
status: OK
-----
Test #9
x=-2.0
calc=-3.626807760141093
ref=-3.626860407847019
rel=1.4516055211840474E-5
status: OK
-----
Test #10
x=5.0
calc=74.20089899580796
ref=74.20321057778875
rel=3.1152048042018 47E-5
status: OK
-----
Test #11
x=-5.0
calc=-74.20089899580796
ref=-74.20321057778875
rel=3.1152048042018 47E-5
status: OK
-----
Test #12
x=10.0
calc=11012.483691068705
ref=11013.232874703393
rel=6.802576892837547E-5
status: OK
-----
Test #13
x=-10.0
calc=-11012.483691068705
ref=-11013.232874703393
rel=6.802576892837547E-5
status: OK
-----
Test #14
x=15.0
calc=1634067.477332553
```

```
ref=1634508.6862359024
rel=2.6993365472136277E-4
status: OK
-----
Test #15
x=-15.0
calc=-1634067.477332553
ref=-1634508.6862359024
rel=2.6993365472136277E-4
status: OK
-----
Test #16
x=20.0
calc=2.4254725709287572E8
ref=2.4258259770489514E8
rel=1.4568486096604497E-4
status: OK
-----
Test #17
x=-20.0
calc=-2.4254725709287572E8
ref=-2.4258259770489514E8
rel=1.4568486096604497E-4
status: OK
-----
Passed: 17 / 17

-- program is finished running (0) --
```

---

## Кросс-проверка на Python

Скрипт валидирует эталон `math.sinh(x)`:

```python
import math

XS = [0.0, 0.001, -0.001, 0.5, -0.5, 1.0, -1.0, 2.0, -2.0, 5.0, -5.0, 10.0,
-10.0, 15.0, -15.0, 20.0, -20.0]

for i, x in enumerate(XS, 1):
    ref = math.sinh(x)
    print("-----")
    print(f"Test #{i}")
```

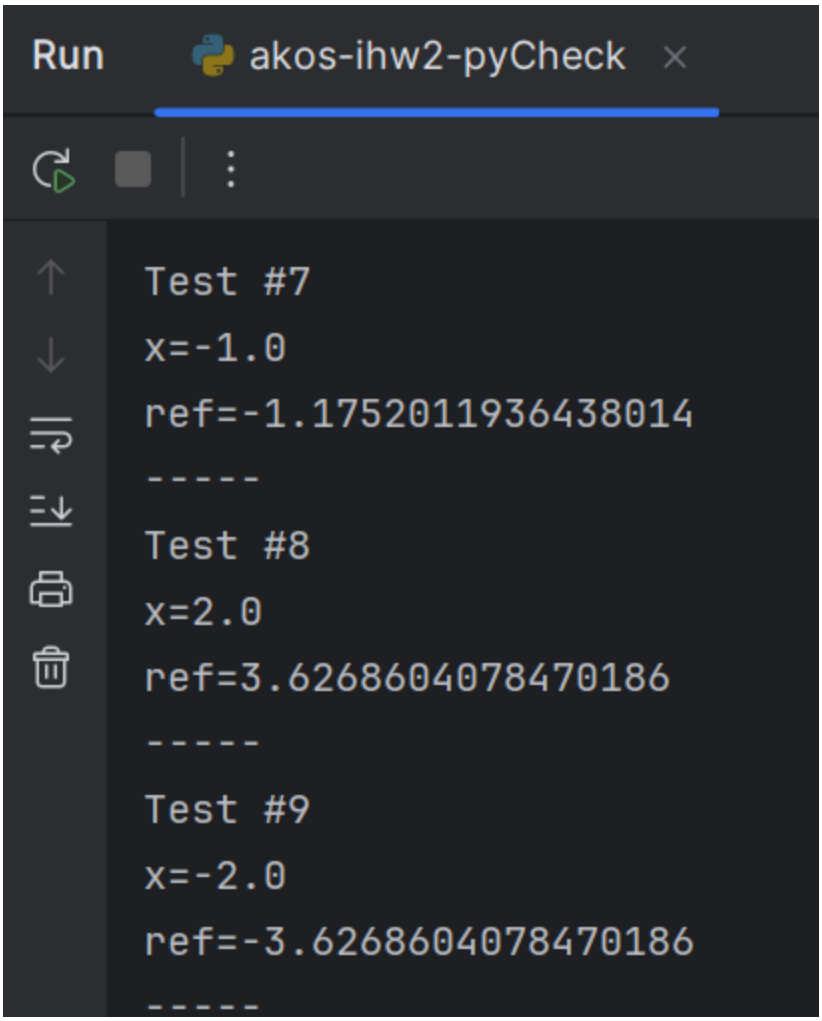
```
    print(f"x={x}")
    print(f"ref={ref}")
```

## Output:

```
-----
Test #1
x=0.0
ref=0.0
-----
Test #2
x=0.001
ref=0.001000000166666675
-----
Test #3
x=-0.001
ref=-0.001000000166666675
-----
Test #4
x=0.5
ref=0.5210953054937474
-----
Test #5
x=-0.5
ref=-0.5210953054937474
-----
Test #6
x=1.0
ref=1.1752011936438014
-----
Test #7
x=-1.0
ref=-1.1752011936438014
-----
Test #8
x=2.0
ref=3.6268604078470186
-----
Test #9
x=-2.0
ref=-3.6268604078470186
-----
Test #10
x=5.0
ref=74.20321057778875
```

```
-----
Test #11
x=-5.0
ref=-74.20321057778875
-----
Test #12
x=10.0
ref=11013.232874703393
-----
Test #13
x=-10.0
ref=-11013.232874703393
-----
Test #14
x=15.0
ref=1634508.6862359024
-----
Test #15
x=-15.0
ref=-1634508.6862359024
-----
Test #16
x=20.0
ref=242582597.70489514
-----
Test #17
x=-20.0
ref=-242582597.70489514
```

## Screen




---

## Изменения по уровням оценки

- **4–5 баллов:**
  - `main.asm` — интерактивный ввод `x` и печать результата.
    (скриншот: ввод значения `x` и вывод `sinh(x)` )
- **6–7 баллов:**
  - Подпрограмма `sinh_series` с параметрами в регистрах ( `fa0/fa1/a0` ), возврат в `fa0` .
  - Локальные переменные на стеке, сохранение `ra` .
- **8 баллов:**
  - Отдельная тестовая программа `main_autotests.asm` без ввода.
  - Набор тестов, покрывающий разные ситуации: `0.0, ±0.001, ±0.5, ±1.0, ±2.0, ±5.0, ±10.0, ±15.0, ±20.0` .
  - Кросс-проверка Python `math.sinh` (скрипт в разделе выше).
- **9 баллов:**
  - Макросы для ввода/вывода и обёртки вычислений ( `macros.asm` ).

## Ограничения и замечания

- Вычисления в double: при очень больших |x| возможны переполнения/медленная сходимость. Тестовый диапазон ограничен **|x| ≤ 20** — все тесты проходят с `rel ≤ 1e-3`.
- Порог `eps_rel = 1e-3` и `max_iter = 4000` подобраны для устойчивости в тестовом наборе.