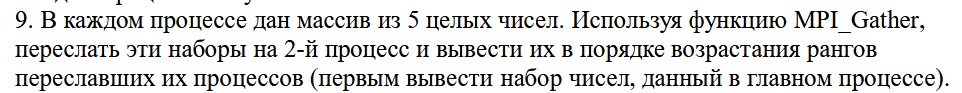
Контрольная №2

Задача на работу с MPI\_Gather

Боязитов Вадим, 932125

25.04.2025

## Задание 1



### Алгоритм

1. Инициализация MPI-окружения

Стандартная инициализация библиотеки MPI

Получение ранга текущего процесса и общего количества процессов

Проверка минимального требуемого количества процессов (>=3). При несоответствии — вывод ошибки и аварийное завершение

2. Подготовка данных

Каждый процесс инициализирует свой локальный массив из 5 элементов

Значения элементов формируются на основе ранга процесса для наглядности

Процесс с рангом 2 (получатель) выделяет память для приёма совокупных данных

Сбор данных с помощью MPI\_Gather

3. Все процессы одновременно вызывают MPI\_Gather

Каждый процесс передаёт свой массив из 5 элементов

Процесс с рангом 2 аккумулирует данные в выделенном буфере

Гарантируется сохранение порядка данных согласно рангам процессов

Вывод результатов

Только процесс с рангом 2 выполняет вывод

Данные выводятся последовательно: от процесса 0 до N-1

Для каждого процесса отображается его ранг и элементы массива

Освобождение выделенной памяти после завершения вывода

4. Корректное завершение

Все процессы синхронно завершают работу с MPI

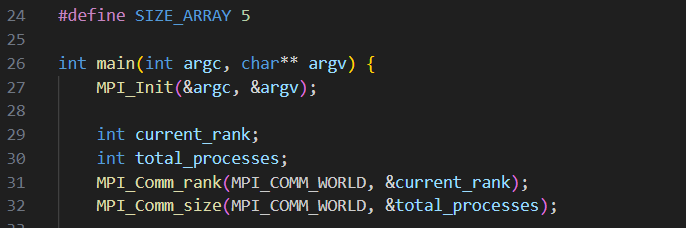
Возврат статуса успешного выполнения программы

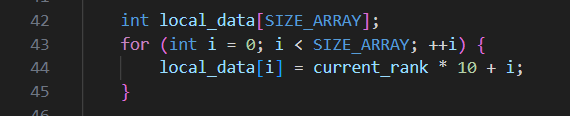
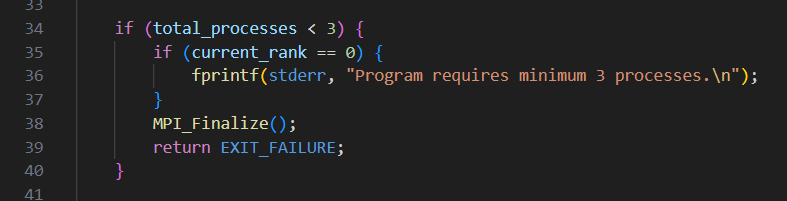
Обработка ошибок памяти и коммуникаций через MPI\_Abort при необходимости

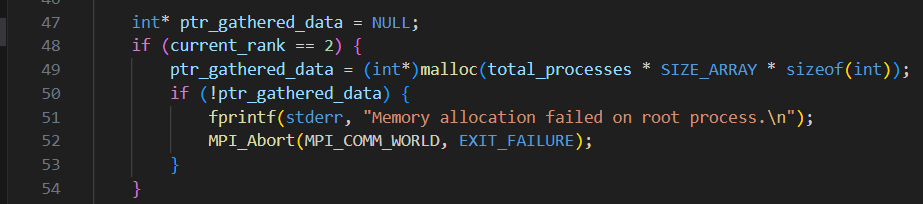
### Программная реализация

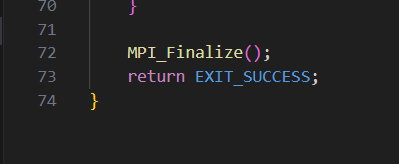
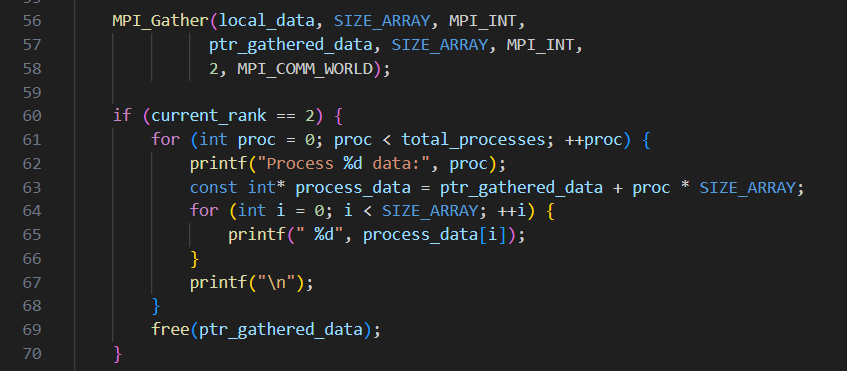
Окружение Linix. Язык C.

<https://github.com/katet3/tsu_parallel_programming/tree/main/lab_2>

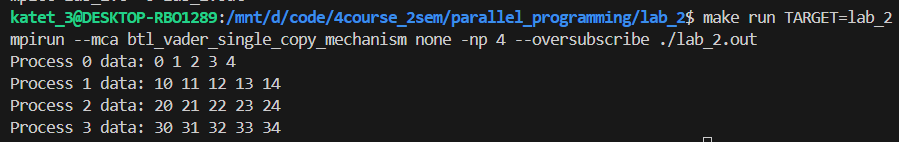








### Тестирование



### Вывод

Получилось путем вызова MPI\_Gather локальные данные каждого процесса, передать в другой процесс.