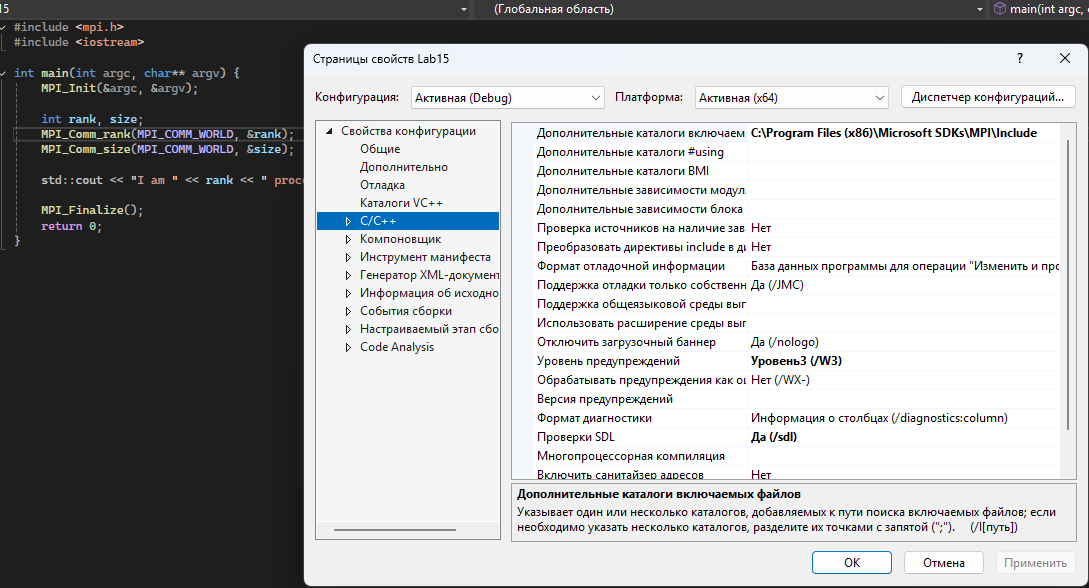
Задание 14. Создание проекта в среде MS Visual Studio с поддержкой MPI

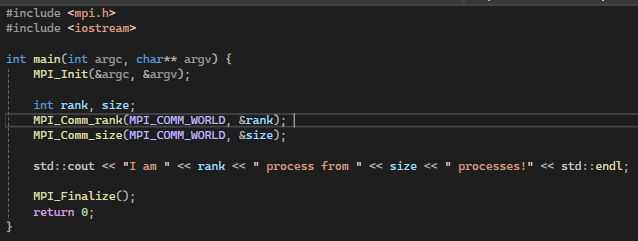
Создайте проект в среде Visual Studio 2010 с поддержкой MPI.

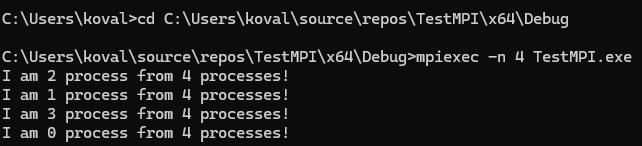


Задание 15. Программа «I am!»

Напишите программу, в которой каждый процесс выводит на экран свой номер и общее количество процессов в приложении в формате:

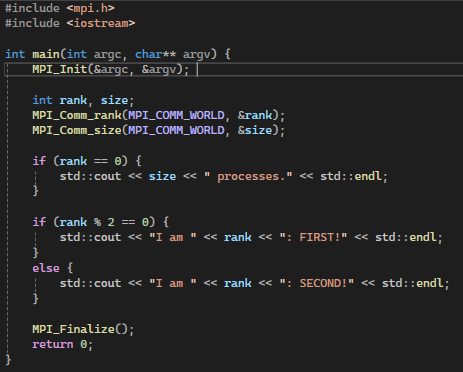
I am <Номер процесса> process from <Количество процессов> processes!

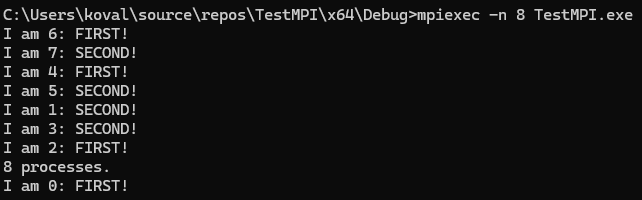




Задание 16. Программа «На первый-второй рассчитайся!»

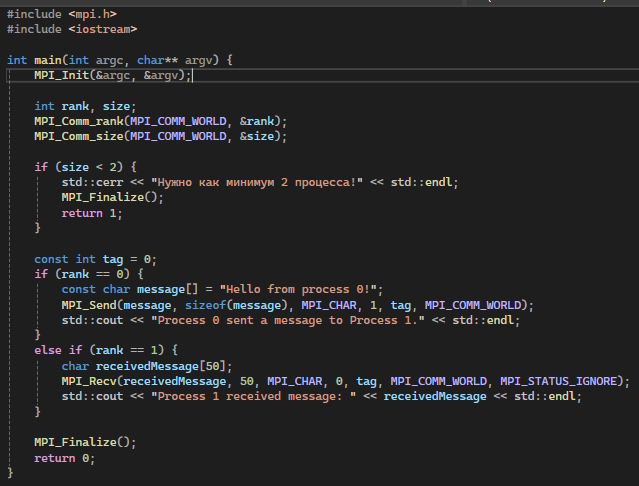
Напишите программу, в которой каждый процесс с четным номером выводит на экран строку «I am <Номер процесса>: FIRST!», а каждый процесс с нечетным номером – «I am <Номер процесса>: SECOND!». Процесс с номером 0 должен вывести на экран общее количество процессов в приложении в формате «<Количество процессов> processes.».

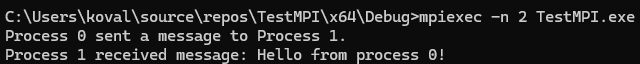




Задание 17. Коммуникации «точка-точка»: простые блокирующие обмены

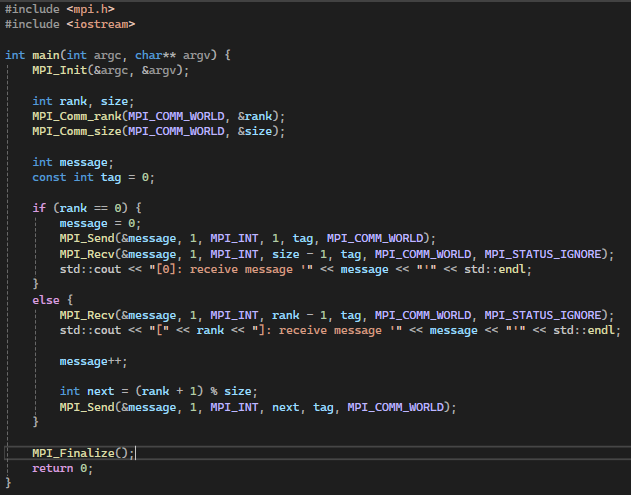
Изучите основные MPI-функции блокирующей передачи сообщений точка-точка MPI\_Send и MPI\_Recv. Напишите MPI-программу, в которой с помощью данных функций процесс с номером 0 отправляет сообщение процессу с номером 1. Процесс 1 выводит полученное сообщение на экран.





Задание 18. Коммуникации «точка-точка»: схема «эстафетная палочка»

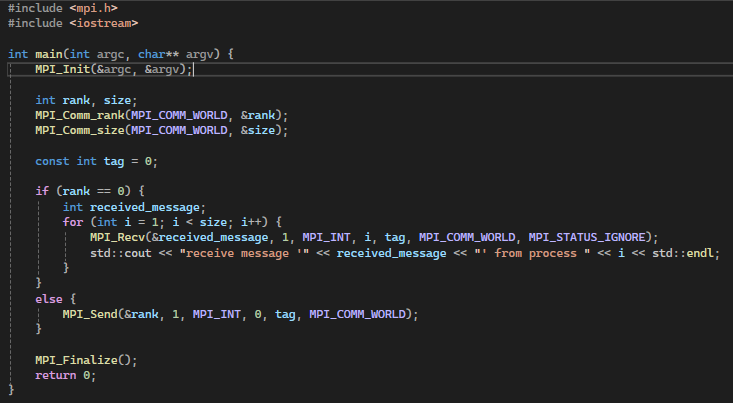
Напишите MPI-программу, реализующую при помощи блокирующих функций посылки сообщений типа точка-точка схему коммуникации процессов «эстафетная палочка», в которой каждый процесс дожидается сообщения от предыдущего и потом посылает следующему (см. рис. 1). В качестве передаваемого сообщения используйте на процессе 0 его номер, на остальных процессах – инкрементированное полученное сообщение.

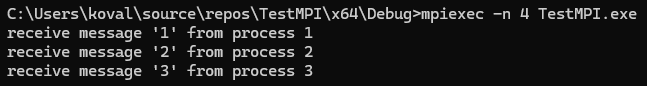




Задание 19. Коммуникации «точка-точка»: схема «мастер-рабочие»

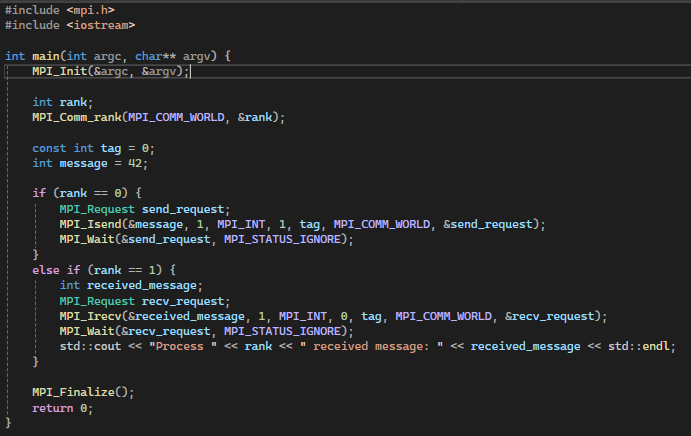
Напишите MPI-программу, реализующую при помощи блокирующих функций посылки сообщений типа точка-точка схему коммуникации процессов «master-slave», в которой один процесс, называемый master , принимает сообщение от остальных процессов, называемых slave. В качестве передаваемого сообщения используйте номер процесса. Master-процесс должен вывести на экран все полученные сообщения.





Задание 20. Коммуникации «точка-точка»: простые неблокирующие обмены

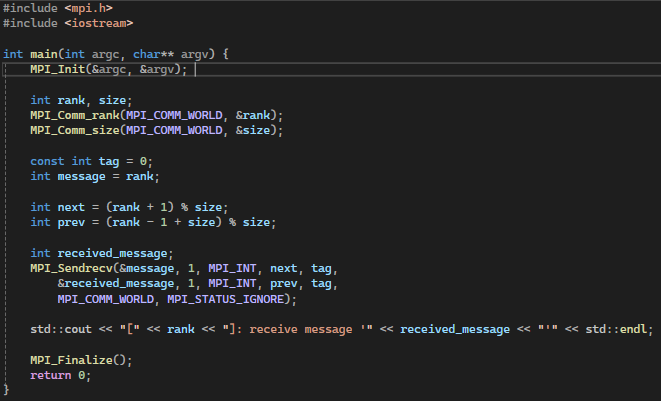
Изучите основные MPI-функции неблокирующей передачи сообщений точка-точка MPI\_Isend, MPI\_Irecv, MPI\_Wait. Напишите MPI-программу, в которой с помощью данных функций процесс с номером 0 отправляет сообщение процессу с номером 1. Процесс 1 выводит полученное сообщение на экран

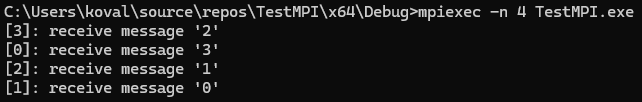




Задание 21. Коммуникации «точка-точка»: схема «сдвиг по кольцу»

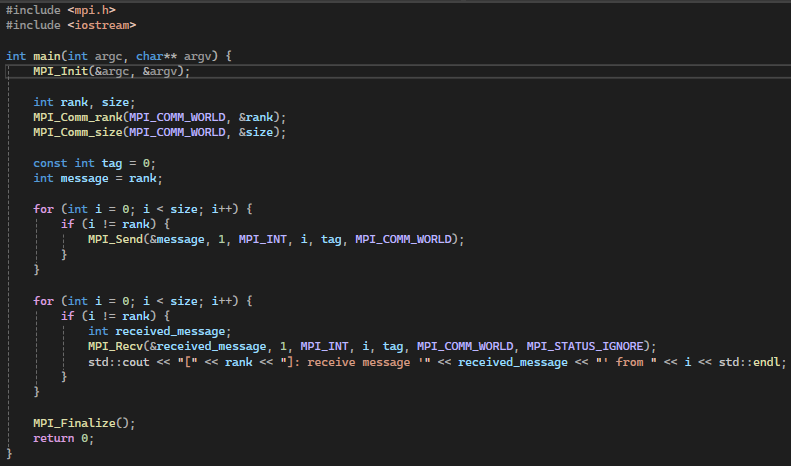
Напишите MPI-программу, реализующую при помощи блокирующих функций посылки сообщений типа точка-точка схему коммуникации процессов «сдвиг по кольцу», в которой осуществляются одновременные посылка и прием сообщений всеми процессами (см. рис. 3). В качестве передаваемого сообщения используйте номер процесса.

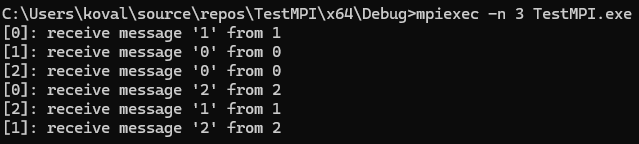




Задание 22. Коммуникации «точка-точка»: схема «каждый каждому»

Напишите MPI-программу, реализующую при помощи блокирующих функций посылки сообщений типа точка-точка схему коммуникации процессов «каждый каждому», в которой осуществляется пересылка сообщения от каждого процесса каждому. В качестве передаваемого сообщения используйте номер процесса. Каждый процесс должен вывести на экран все полученные сообщения.

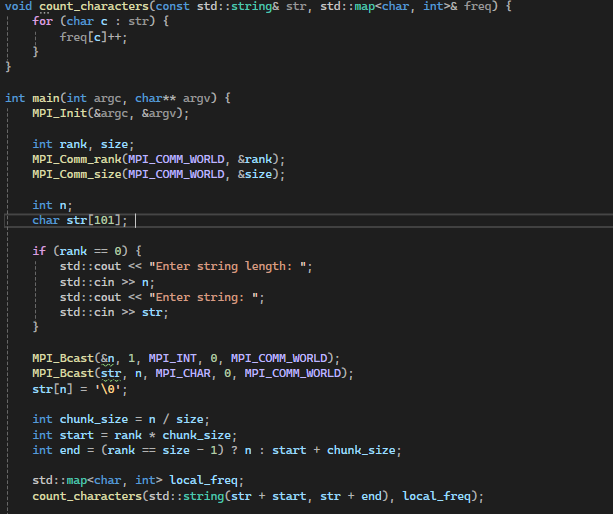


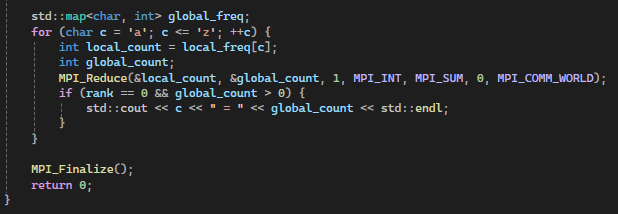


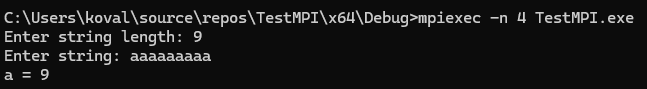
Задание 23. Коллективные коммуникации: широковещательная рассылка

данных

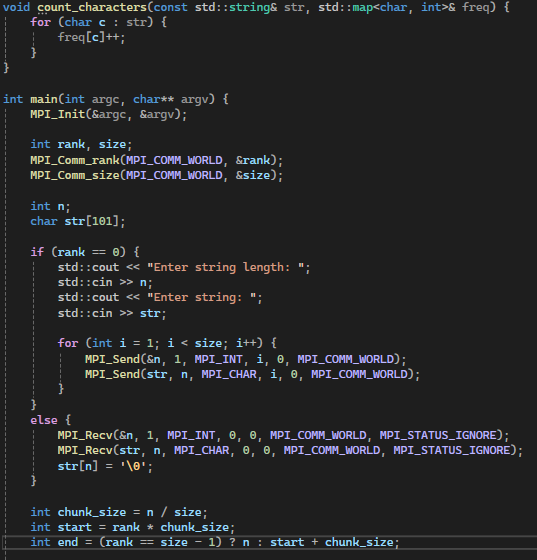
1. Изучите MPI-функцию широковещательной рассылки данных MPI\_Bcast. Напишите MPI-программу, которая в строке длины n определяет количество вхождений символов. Ввод данных должен осуществляться процессом с номером 0. Для рассылки строки поиска и ее длины по процессам используйте функцию MPI\_Bcast.

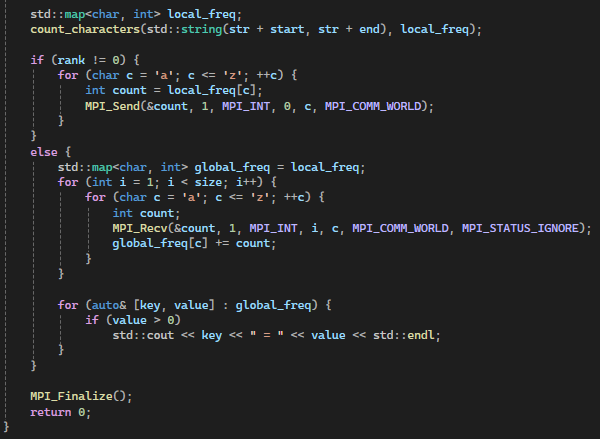


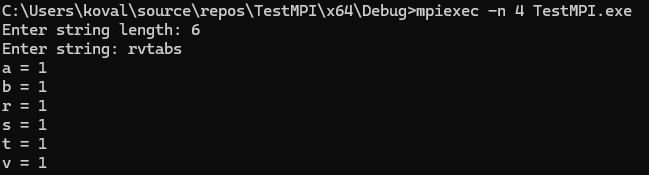




2\*. Перепишите программу, используя вместо функции MPI\_Bcast функции коммуникации «точка-точка». Сравните эффективность выполнения программ с коллективными и точечными обменами.







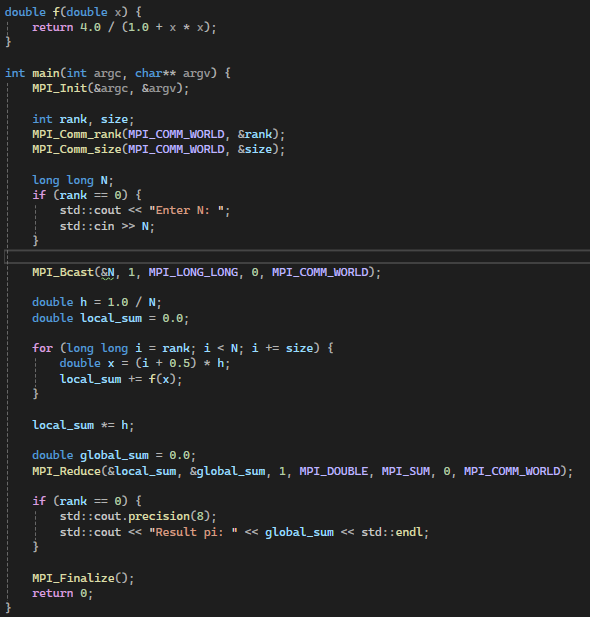
MPI\_Bcast – эффективен для рассылки одного сообщения всем процессам

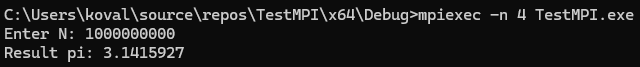
MPI\_Send/Recv – гибкость при отправке индивидуальных данных

Вывод: MPI\_Bcast эффективнее в данном случае, так как позволяет передавать строку всем процессам одновременно

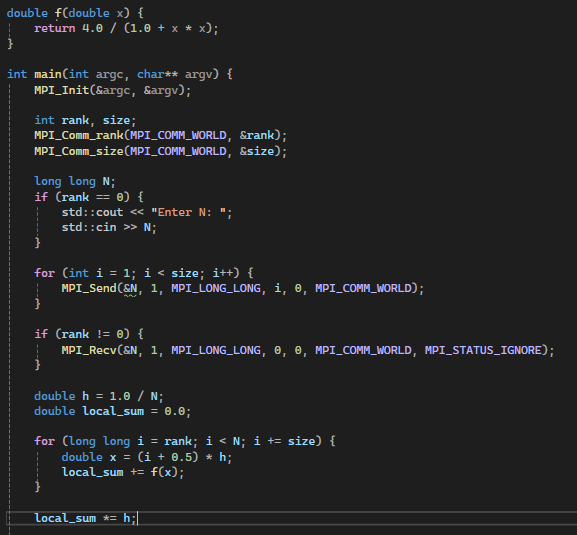
Задание 24. Коллективные коммуникации: операции редукции

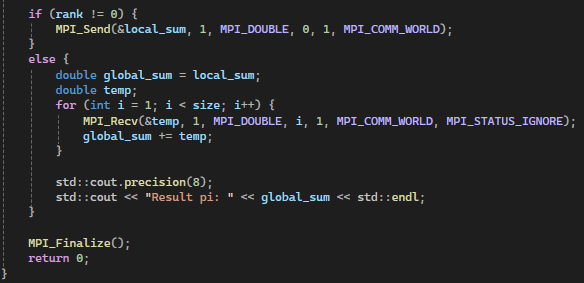
1. Изучите MPI-функцию для выполнения операций редукции над данными, расположенными в адресных пространствах различных процессов, MPI\_Reduce. Реализуйте программу вычисления числа 𝜋 (см. задание 8), используйте функцию MPI\_Reduce для суммирования результатов, вычисленных каждым процессом.





2\*. Перепишите программу, используя вместо функции MPI\_Reduce функции коммуникации «точка-точка». Сравните эффективность выполнения программ с коллективными и точечными обменами.



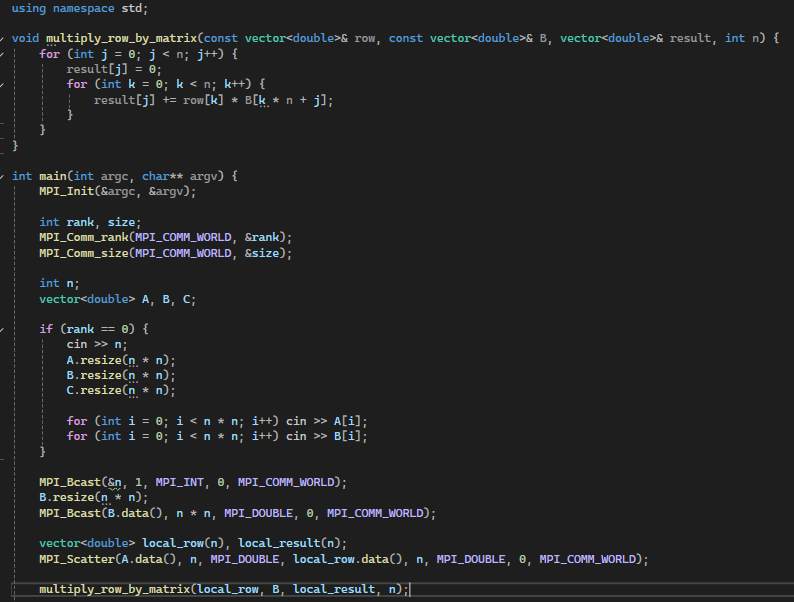


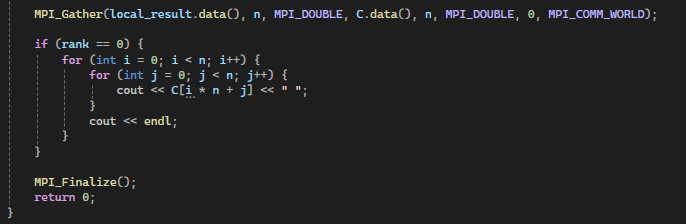


MPI\_Reduce – быстрее, так как использует оптимизированные схемы суммирования данных

Задание 25. Коллективные коммуникации: функции распределения и сбора данных

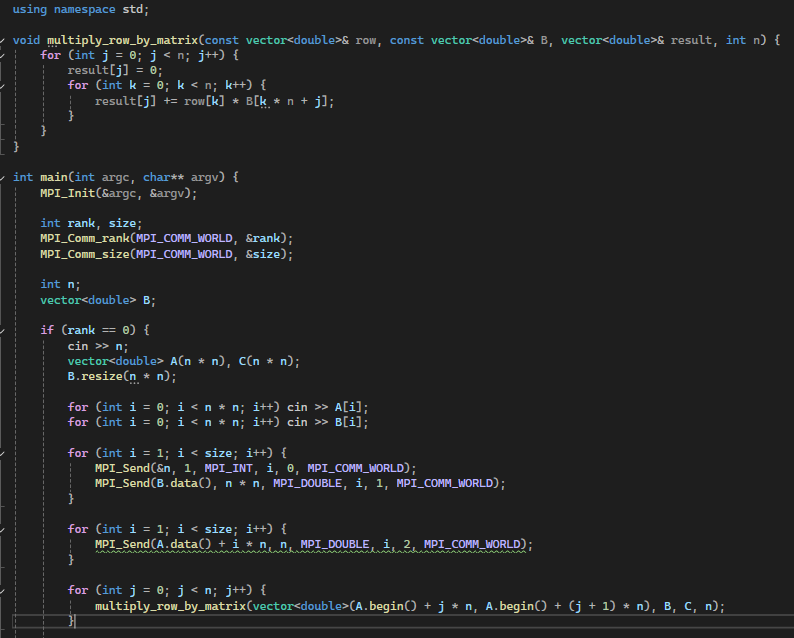
1. Изучите MPI-функции распределения и сбора блоков данных по процессам MPI\_Scatter и MPI\_Gather. Напишите программу, которая вычисляет произведение двух квадратных матриц 𝐴 × 𝐵 = С размера 𝑛 × 𝑛. Используйте формулу, приведенную в задании 9. Ввод данных и вывод результата должны осуществляться процессом с номером 0. Для распределения матриц A и B и сбора матрицы С используйте функций MPI\_Scatter и MPI\_Gather.

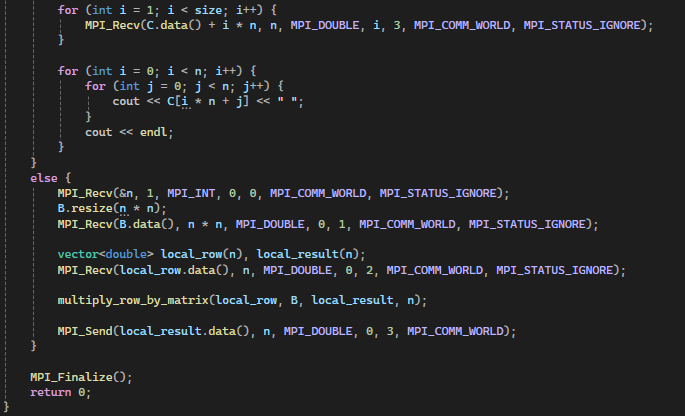


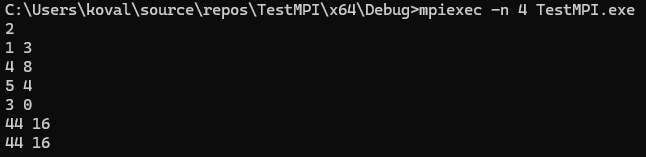




2\*. Перепишите программу, используя вместо функций MPI\_Scatter и MPI\_Gather функции коммуникации «точка-точка». Сравните эффективность выполнения программ с коллективными и точечными обменами.



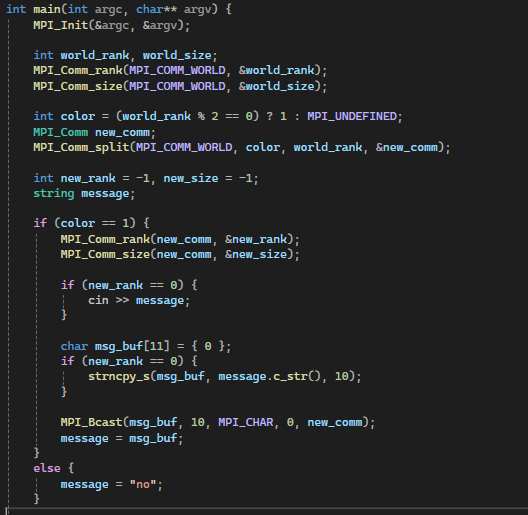


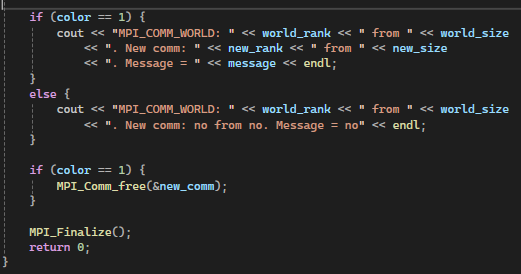


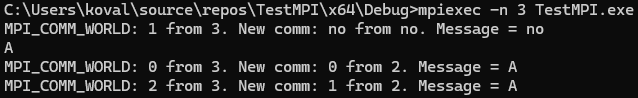
MPI\_Scatter/MPI\_Gather быстрее, так как снижает количество точечных сообщений

Задание 26. Группы и коммуникаторы

Напишите программу, в которой производится широковещательная рассылка сообщения с помощью функции MPI\_Bcast, но только по процессам с четным номером. Для рассылки сообщения создайте новый коммуникатор. Каждый процесс приложения должен выводить на экран «MPI\_COMM\_WORLD: <номер процесса в коммуникаторе MPI\_COMM\_WORLD> from <количество процессов в коммуникаторе MPI\_COMM\_WORLD>. New comm: <номер процесса в новом коммуникаторе> from <количество процессов в новом коммуникаторе>. Message = <сообщение>»

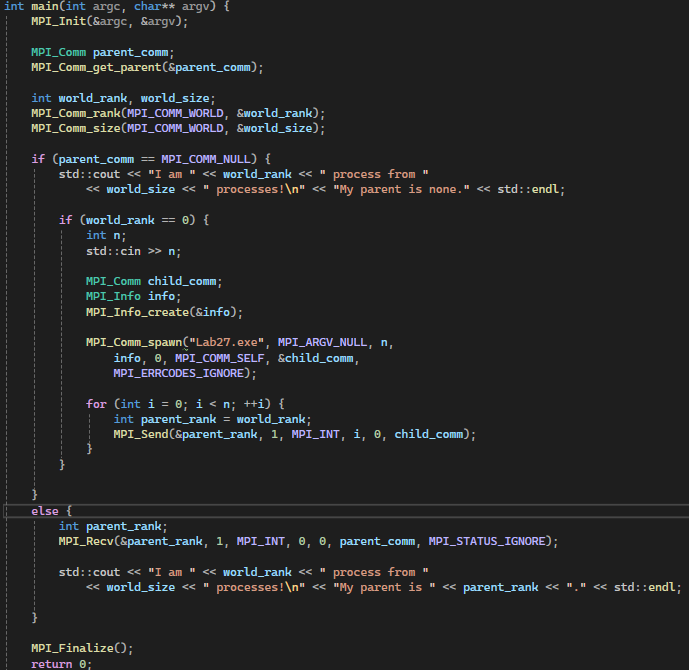


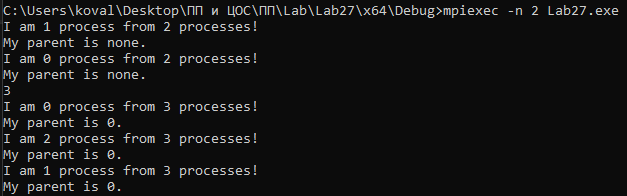




Задание 27\*. MPI-2: динамическое создание процессов

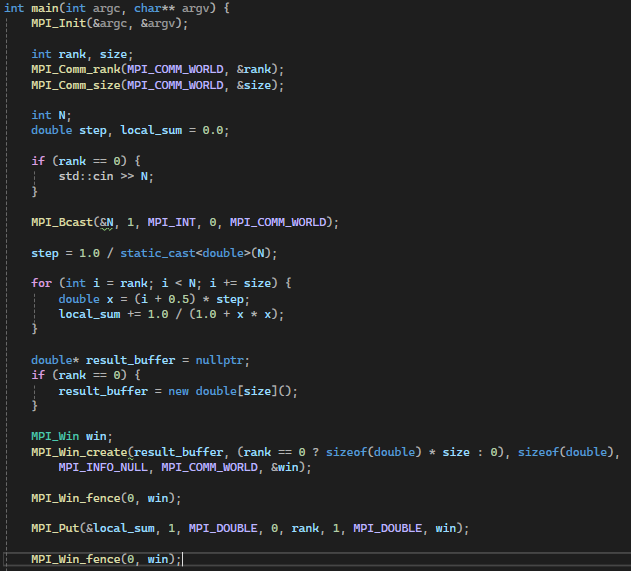
Напишите программу, в которой нулевым процессом динамически запускается еще n процессов. Каждый процесс в программе выводит сообщение в формате: «I am <Номер процесса> process from <Количество процессов> processes! My parent is <Номер процесса родителя>».

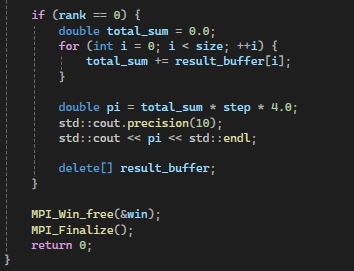




Задание 28\*. MPI-2: односторонние коммуникации

Реализуйте программу вычисления числа 𝜋 (см. задание 24), используйте функции односторонней коммуникации для обмена данными между процессами.







Задание 29. Исследование масштабируемости MPI-программ

Проведите серию экспериментов на суперкомпьютере по исследованию масштабируемости OpenMP-программ. Заполните таблицу.

Обновленный код задания 24:

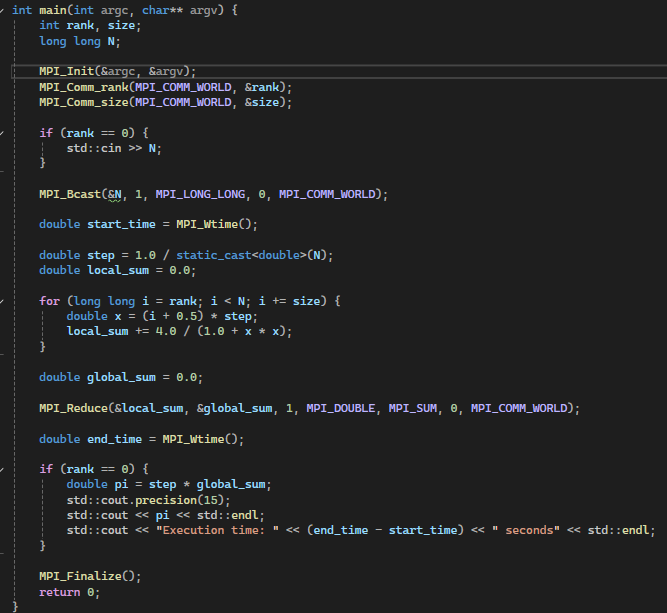
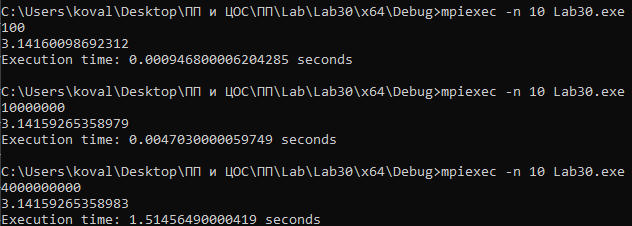
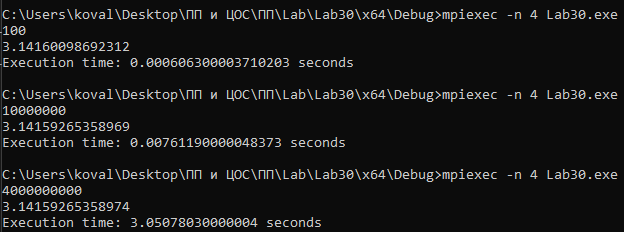


Таблица:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | Программа | Параметр N | Количество нитей | Время выполнения  (сек.) |
| 57 | «Число 𝜋» | 100 | 1 | 0.00017 |
| 58 | 10 000 000 | 1 | 0.027 |
| 59 | 4 000 000 000 | 1 | 11.201 |
| 60 |  | 100 | 2 | 0.00038 |
| 61 | 10 000 000 | 2 | 0.014 |
| 62 | 4 000 000 000 | 2 | 6.088 |
| 63 |  | 100 | 4 | 0.0006 |
| 64 | 10 000 000 | 4 | 0.0076 |
| 65 | 4 000 000 000 | 4 | 3.05 |
| 66 |  | 100 | 6 | 0.0006 |
| 67 | 10 000 000 | 6 | 0.0059 |
| 68 | 4 000 000 000 | 6 | 2.13 |
| 69 |  | 100 | 8 | 0.0008 |
| 70 | 10 000 000 | 8 | 0.0057 |
| 71 | 4 000 000 000 | 8 | 1.75 |
| 72 |  | 100 | 10 | 0.0009 |
| 73 | 10 000 000 | 10 | 0.0047 |
| 74 | 4 000 000 000 | 10 | 1.51 |
| 75 |  | 100 | 12 | 0.0009 |
| 76 | 10 000 000 | 12 | 0.0044 |
| 77 | 4 000 000 000 | 12 | 1.36 |

Некоторые резаультаты:

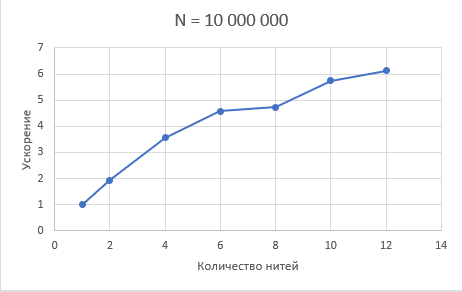




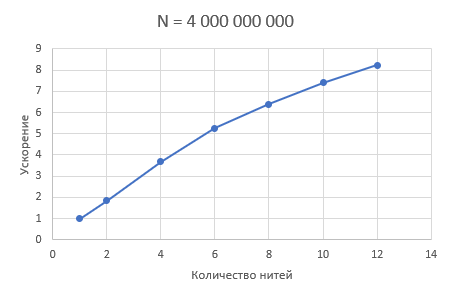
На основании данных таблицы постройте график масштабируемости для каждого значения параметра N. Определите для каждого графика, при каком количестве нитей достигается максимальное ускорение.



Максимальное ускорение = 1, при количестве нитей = 1



Максимальное ускорение = 6.136, при количестве нитей = 12



Максимальное ускорение = 8.23, при количестве нитей = 12

Задание 30. Проект в среде Visual Studio 2010 с поддержкой MPI и OpenMP

Создайте проект в среде Visual Studio 2010 с минимальным кодом с поддержкой MPI и OpenMP.



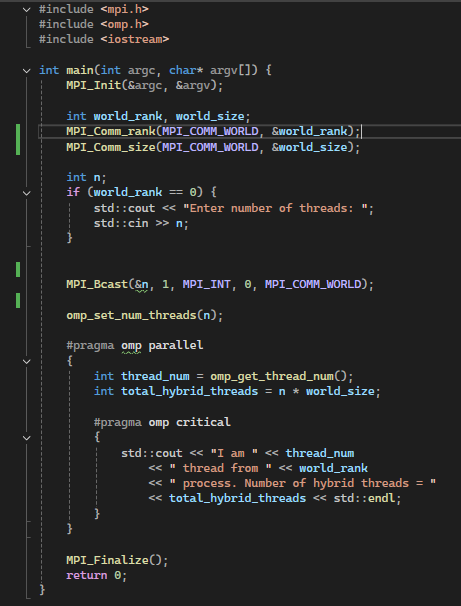


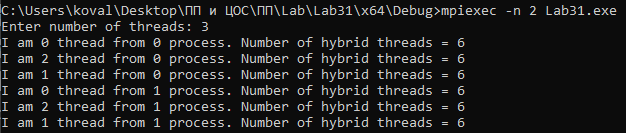




Задание 31. Программа «I am»

Напишите программу, в которой в каждом процессе создается n нитей. Каждая нить должна выводить на экран свой номер, номер процесса родителя и общее количество нитей во всех процессах в следующем формате: I am <Номер нити> thread from <Номер родительского процесса> process. Number of hybrid threads = <Количество нитей \* Количество процессов>.





Задание 32. Программа «Число 𝜋»

Реализуйте программу вычисления числа 𝜋 (см. задание 8) с использованием MPI+OpenMP.

