1.1

В программе используются базовые функции MPI: инициализация и окончание параллельной части (соответственно *MPI\_Init* и *MPI\_Finalize*), а также получения числа процессов и номера конкретного процесса (соответственно *MPI\_Comm\_size* и *MPI\_Comm\_rank*).

1.2

В программе вводится простейшая операция передачи сообщений между потоками – блокирующий обмен *MPI\_Send*/*MPI\_Recv*. Программа работает по следующему алгоритму:

1. На каждом из потоков создается буфер (массив) message. На нулевом потоке он заполняется сообщением.
2. С помощью *MPI\_Send* на первый поток из нулевого отправляется 20 элементов буфера с сообщением.
3. На первом потоке MPI\_Recv блокирует выполнение кода до принятия; сообщение принимается в message и выводится на экран.

1.3

В программе используются те же самые конструкции, что и в предыдущей. Каждый поток проходит проверку:

1. Если поток четный, он отправляет свой номер потока (rank) потоку следующему потоку (rank + 1).
2. Если поток нечетный, то он принимает сообщение и выводит на экран.

2.1

2.2

В программе используются вызовы программы пробника *MPI\_Probe*. Программа работает по следующему алгоритму:

1. На нулевом процессе массив заполняется числами и отправляется на первый процесс операцией *MPI\_Send*. Размер массива, в качестве примера, равен 5, но возможны другие значения.
2. На первом потоке *MPI\_Probe* принимает сообщение, но не записывает в буфер. Информация о сообщение записывается в ранее инициализированную структуру *status*. С помощью функции *MPI\_Get\_count* из status извлекается информация о количестве принимаемых элементов.
3. Создается массив, память под который выделяется динамически и равна 5 \* *sizeof(тип данных)*. Таким образом получается избежать ошибок с памятью.

2.3