

4 балла

25. **Задача о производстве булавок.** В цехе по заточке булавок все необходимые операции осуществляются **тремя** рабочими. Первый из них берет булавку и проверяет ее на предмет кривизны. Если булавка не кривая, то рабочий передает ее своему напарнику. Напарник осуществляет собственно заточку и передает заточенную булавку третьему рабочему, который осуществляет контроль качества операции. *Требуется создать многопоточное приложение, моделирующее работу цеха. При решении использовать парадигму «производитель-потребитель».* Следует учесть, что каждая из операций выполняется за случайное время которое не связано с конкретным рабочим. Возможны различные способы передачи (на выбор). Либо непосредственно по одной булавке, либо через ящики, в которых буферизируется некоторое конечное количество булавок.

- В данной задаче представлена реализация модели: Производители и потребители.

Производители и потребители – это парадигма взаимодействующих неравноправных потоков. Одни потоки «производят» данные, другие их «потребляют». Данная модель подразумевает использование блокировок чтения-записи. Их использование здесь актуально, ведь одновременно работают несколько потоков читателей.

В реализации данной задачи также используются мьютексы. Они необходимы для реализации последовательного вывода информации в консоль (файл), т. е. единоличного доступа к данному ресурсу.

- Единственным входным параметром в данной задаче является размер массива (количество рассматриваемых булавок). Его можно задать 3 способами:
 - Консоль – ввод целого положительного числа в консоль
 - Файл – ввод значения из указанного в командной строке файла целого положительного числа
 - Случайная генерация – генерация длины массива в интервале от (2; 7)
- Реализация игры на C++ 17 заключается в следующем:

Для каждого из трёх работников есть два потока – поток читатель и поток писатель. Сначала работник потоком писателем даёт оценку булавке (может ли она идти к следующему работнику или нет) одной из доступных для него булавок (`worker1Vec`). Далее потом читателем он считывает одно из оцененных им значений и в зависимости от заданного каждому работнику интервала, решает, передать булавку (индекс в общем массиве) другому работнику или оставить у себя и произвести оценку ещё раз. В конце, когда все индексы массива принадлежат 3 работнику, т. е. все булавки были осмотрены, то программа завершает свою работу.

В таблице ниже приведены значения интервалов случайной генерации оценки, а также допустимый интервал “приёмки” булавки и передачи её дальше:

	Генерация	Допустимый интервал
worker 1	[5; 20]	[10; 20]
worker 2	[15; 30]	[20; 30]
worker 3	[35; 60]	[40; 60]

5 баллов

- Программа на C++ снабжена необходимыми комментариями
- Взаимодействие «сущностей» описано выше в пункте на 4 балла

6 баллов

- Алгоритм работы программы заключается во взаимодействие функций данной программы. В функции `main` создаются 3 потока читателя и 3 потока писателя. Далее взаимодействие с ними происходит за счёт функций **funcRead** и **funcWrite**, которые разбиваются на более мелкие функции по взаимодействию с отдельными потоками, как бы процессами работы отдельного работника. В программе также есть функции для случайной генерации значений в интервале (a; b) и три функции по выводу информации в консоль (опционально – в файл `output.txt`)
- В командную строку вводится путь к входному файлу.

7 баллов

- В командную строку вводится путь к файлу и из него считывается размер массива. Вывод результатов происходит в файл output.txt
- К данной работе прикреплены файлы input_2.txt и output_2.txt; input_3.txt и output_3.txt; input_4.txt и output_4.txt с соответствующими значениями длины массива 2, 3 или 4

8 баллов

- Случайная генерация – генерация длины массива в интервале от (2; 7)
- Расширение параметров командной строки не предусмотрено, так как генерация происходит без использования параметра seed, а за счёт std::random_device из библиотеки *random*
- Примеры выходных в консоль данных при случайной генерации

```

Type in the console the type of input you want:
1 - console (output in console)
2 - file input (output in output.txt + console)
3 - random input (output in console)
Input type:..

The random size is 5
Worker 1 examined Element[4] -> 5
Element[0] -> 1 failed examination by worker-reader 1
Worker 1 examined Element[4] -> 12
Worker 1 examined Element[3] -> 5
Worker 1 examined Element[1] -> 12
Element[2] -> 9 failed examination by worker-reader 1
Worker 1 examined Element[1] -> 13
Worker 1 examined Element[1] -> 13
Element[0] -> 1 failed examination by worker-reader 1
Worker 1 examined Element[3] -> 16
Worker 1 examined Element[3] -> 16
Element[2] -> 9 failed examination by worker-reader 1
Worker 1 examined Element[3] -> 13
Worker 1 examined Element[1] -> 12
Worker 1 examined Element[1] -> 10
Worker 1 examined Element[4] -> 12
Element[2] -> 9 failed examination by worker-reader 1
Worker 1 examined Element[0] -> 18
Worker 1 examined Element[0] -> 18
Element[3] -> 13 passed examination by worker-reader 1
Worker 2 examined Element[3] -> 16
Worker 1 examined Element[0] -> 13
Worker 1 examined Element[1] -> 12
Worker 2 examined Element[3] -> 23
Worker 1 examined Element[4] -> 20
Worker 1 examined Element[0] -> 7
Element[3] -> 23 passed examination by worker-reader 2
Element[0] -> 7 failed examination by worker-reader 1
Worker 3 examined Element[3] -> 52
Worker 1 examined Element[2] -> 15
Worker 1 examined Element[0] -> 13
Element[3] -> 52 passed examination by worker-reader 3
Element[4] -> 20 passed examination by worker-reader 1
Worker 2 examined Element[4] -> 24
Worker 1 examined Element[2] -> 19
Worker 1 examined Element[2] -> 19
Element[4] -> 24 passed examination by worker-reader 2
Element[4] -> 24 failed examination by worker-reader 3
Element[0] -> 13 passed examination by worker-reader 1
Worker 3 examined Element[4] -> 55
Worker 2 examined Element[0] -> 25
Worker 1 examined Element[1] -> 9
Worker 1 examined Element[2] -> 7
Worker 3 examined Element[4] -> 44
Worker 2 examined Element[0] -> 26
Worker 1 examined Element[1] -> 6
Worker 1 examined Element[1] -> 6
Element[0] -> 26 passed examination by worker-reader 2
Element[4] -> 44 passed examination by worker-reader 3
Element[1] -> 6 failed examination by worker-reader 1
Worker 3 examined Element[0] -> 57
Worker 1 examined Element[1] -> 9
Worker 1 examined Element[1] -> 9
Element[0] -> 57 passed examination by worker-reader 3
Element[1] -> 9 failed examination by worker-reader 1
Worker 1 examined Element[2] -> 10
Worker 1 examined Element[1] -> 15
Element[2] -> 10 passed examination by worker-reader 1
Worker 2 examined Element[2] -> 16
Worker 1 examined Element[1] -> 16
Worker 1 examined Element[1] -> 16
Worker 2 examined Element[2] -> 17
Worker 1 examined Element[1] -> 10
Worker 1 examined Element[1] -> 10

```

```

Element[2] -> 17 failed examination by worker-reader 2
Element[1] -> 10 passed examination by worker-reader 1
Worker 2 examined Element[1] -> 29
Element[2] -> 17 failed examination by worker-reader 2
Worker 2 examined Element[1] -> 19
Element[2] -> 17 failed examination by worker-reader 2
Worker 2 examined Element[2] -> 26
Worker 2 examined Element[1] -> 28
Element[2] -> 26 passed examination by worker-reader 2
Element[2] -> 26 failed examination by worker-reader 3
Worker 3 examined Element[2] -> 52
Worker 2 examined Element[1] -> 30
Element[1] -> 30 passed examination by worker-reader 2
Element[1] -> 30 failed examination by worker-reader 3
Worker 3 examined Element[2] -> 56
Element[1] -> 30 failed examination by worker-reader 3
Worker 3 examined Element[1] -> 36
Worker 3 examined Element[2] -> 59
Element[1] -> 36 failed examination by worker-reader 3
Worker 3 examined Element[2] -> 39
Element[1] -> 36 failed examination by worker-reader 3
Worker 3 examined Element[2] -> 55
Element[1] -> 36 failed examination by worker-reader 3
Worker 3 examined Element[1] -> 49
Worker 3 examined Element[2] -> 57
Element[1] -> 49 passed examination by worker-reader 3
Worker 3 examined Element[2] -> 36
Element[2] -> 36 failed examination by worker-reader 3
Worker 3 examined Element[2] -> 44
Element[2] -> 44 passed examination by worker-reader 3

Process has finished!

```

Данный «журнал событий», «лог» того как работают потоки при значении размера массива 5 является достаточно большим. Поэтому случайная генерация производится до значения 7.