Задание №3:

1) Написать свою реализацию встроенной функции массивов filter. Назвать функцию myFilter и сделать так, чтобы любой массив мог использовать данную функцию как "родную". В качестве параметров он должен принимать callback-функцию и как необязательный параметр объект, который будет использован в качестве this в рамках внутренних вызовов данной callback-функции.
В конечном итоге ваша реализация myFilter должна работать точно также как и встроенный метод filter. Callback-функция, переданная в качестве параметра, также должна вызываться с теми же параметрами, что и оригинал (элемент, индекс, массив).

2) Написать функцию createDebounceFunction. Она должна принимать два аргумента: callback-функцию и задержку в миллисекундах. Данная функция должна возвращать новую функцию, вызывающую callback-функцию с задержкой в переданное количество миллисекунд. ПРИ ЭТОМ! Если за то время, пока внутренння callback-функция ждёт своего вызова, наша созданная функция вызывается ещё раз, то "счётчик" времени должен сбрасываться и начинаться заново (т.е. вызова внутренней функции произойти не должно).

> Пример:
```
const log100 = () => console.log(100);
const debounceLog100 = createDebounceFunction(log100, 1000);
debounceLog100();
setTimeout(debounceLog100, 200); // так как задержка в 1000мс и новый вызов этой же функции происходит через 200 миллисекунд, то таймер запускается заново
setTimeout(debounceLog100, 400); // снова сбрасываем таймер ещё через 200 миллисекунд
```

Вывод в консоли значения 100 примерно через 1000мс + 200мс + 200мс