2) Если это чисто виртуальная функция, то это error. так сделать нельзя, еще в момент компиляции выдаст ошибку, но можно засунуть вызов чисто виртуальной функции в обычную функцию, тогда ошибки не будет. Если же это обычная виртуальная функция, то все успешно запуститься.

3) Разница в том, что по умоланию поля класса private, а у структуры public.

4) malloc - выделяет нужну память в куче, а new - выделяет и размещает обьект в этой выделенной памяти.

5) delete - удаляет обьект выделенный на динамически, а delete[] удаляет массив обьектов выделлынх на куче.

6) Разница в том как они хранят обьекты, list хранит указатель на следующий и предыдущий элемент, что дает нам создание нового элемента и удаление первого и последнего элемента за O(1), если мы собираемся искать что-то, например 7 элемент в листе и достать оттуда занчение, то придется пробежать начиная с первого до 7 элемента, что означает слодность O(n); vector хранит все свои обьекты рядом, если у него заканчивается capacity, то он выделяет новое место побольше в куче, все переносит туда, по этому вставка в vector, иногда может быть проблематична по времени, зато выбор по индексу происходит за O(1). По этому если вы часто добавляете, удаляете и не ищете по индексу элементы то выбираем list, если у вам изначальна данно много обьектов, и вам нужно их вытаскивать по индексу, то выбираем vector.

7) std::auto_ptr - относсится к умным указателям использующем идиому RAII, но он устарел и не рекомендуется к парктике, вместо него используйте std::uniq_ptr или std::shared_ptr.

8) Да, просиходит копирование, так как "a" - это l value обект.

9) Да, происходит копирование, и это возможно, потому что класс qstring имеет перегруженный конструктор(const char *str)

10) Нет, так как при таком подходе в обьекте clist будут висчие указатели, из-за того что в цикле мы ссылаемся на temporary object QByteArray, вызываемый функцией toLocal8Bit(), который в коце цикал заканчивает жить.

11)

* Процесс - это независимый экземпляр программы,для которго выделятся системные ресурсы. Один процесс не может получить доступ в адрессную память другуго процесса. Поток - это последовательность комманд, работающих в рамках некоторго процесса.

* Методы синхронизации потоков - если мы говорим о том, что бы все потоки спокойно работали и не случалось состоянии гонки, то с этим нам поможет: std::mutex(и его разновиднсти), std::scoped_lock(и так же его разновидности), std::atomic. Если же гоорвить о синхронизации как о том, что все потоки в какой-то чати программы должны встать на линию и ждать других поток(как будто на стартовой точке, чтобы потом опять ринутсся), то с этим нам поможет std::launch или std::barrier.

* thread safe class - это класс, в котором методы могут быить безапасно вызваны разным количеством потоков, изпользуя один и тот же экземпляр обекта.

12) Сложность алгоритмов оценивается по двум параметрам: по времени выполнения и по используемой памяти. Обычно используют обреваитуру O(*), где * - сложность. Сложность высчитывается сколько раз нам нужно пройти контейнер, чтобы достичь своей цели - это для времени выполнения, а по памяти - это сколько раз нам нужно выделить такого размера как исходный контейнер памяти.

* std::vector : insert - O(n), так как элементы слева от вставки нужно будет сдвинуть влево; find - O(n), так как нужно пройти все элементы, чтобы найти конкретный

* std::list : insert - O(1), так как при в ставке нам нужно лишь поменять указатели слева и справа; find - O(n), так как нужно пройти все элементы, чтобы найти конкретный

* std::map : insert - O(log N) - так как происходит вставка по бинарному дерево, find() аналогично.

* std::vector sort : insert - O(n), ничего нового, find - здесь взавиимости от какого способо поиска, обычного O(n) или бинарного O(n)

* std::list sort : insert - O(n), ничего нового, find - не рекомендуется использовать бинарный поиск с листом, так как это проблематично и не особо эффективно, по этому опять получаем O(n)

13)

```
select p.name

from Parents as p

inner join

Kids as k

on p.id = k.parent_id

group by p.id

having count(k.id) <= 2
```

14) 8 байт, так как русские символы занимают 2 байта.