1. **Дайте определение записи со статическими полями**

Представляет собой одну или несколько переменных, которые объединены под один именем

2. **Определён ли порядок инициализации полей при объявлении записи?**

Да

Для инициализации переменной структурного типа необходимо указать список значений, заключенный в фигурные скобки. Значения в списке должны появляться в том же порядке, что и имена полей структуры. Если значений меньше, чем полей структуры, оставшиеся поля инициализируются нулями.

В стандарте C99, можно задавать структуры следующим образом. Тогда задавать значения можно в произвольном порядке

```
struct date exam = {.day = 13, .month = 1, .year = 2019};
struct date exam = {.day = 13, 1};
```

3. **Как можно обратиться к полю записи и полю записи под указателем?**

```
(*some_struct).field = 3;
some_struct->field = 3;
```

4. **Что происходит при присваивании записей?**

При присваивании структуры происходит перекопирование всех полей одной структуры в другую

5. **Дайте определение выравниванию полей записи в памяти. Кто осуществляет выравнивание? Какова причина?**

Выравнивание - размещение полей структуры в памяти так, чтобы адрес был кратен размеру поля

Выравнивание осуществляет транслятор

Причина: увеличение скорости доступа, улучшение производительности программы

6. **Как можно определить объём занимаемой записью памяти?**

С помощью sizeof()

7. **Дайте определение упаковке записей.**

Упаковка - выравнивание памяти для полей записей таким образом, чтобы минимизировать использование памяти. При упаковке компилятор игнорирует или сокращает промежутки между полями записи, чтобы поля располагались непрерывно, без дополнительных байтов для выравнивания.

8. **Есть ли упоминание упаковки записей в стандарте?**

Нет, есть только упоминание директивы #pragma

9. **В каких случаях рекомендуется пользоваться упаковкой и почему?**

В случаях, когда важно экономить память

10. **Какова мотивация к использованию массива записей перед использованием совокупности массивов?**

Удобство доступа, обработки информации, однозначность данных

11. **Дайте определение последовательному доступу к информации. Чем он отличается от произвольного доступа?**

Последовательный доступ к информации - информация записывается/читается друг за другом

Произвольный доступ к информации - информацию можно читать/записывать в произвольном порядке, без предварительной обработки информации расположенной до интересующего участка

12. **Дайте определение физическому файлу.**

Именовая область на диске

13. **Дайте определение сектору и сегменту.**

Сектор – минимальная адресуемая единица хранения информации на дисковых запоминающих устройствах

Сегмент – область памяти процесса или программы, которая может быть выделена или освобождена независимо друг от друга // условно выделенная область адресного пространства определённого размера

14. **Можно ли провести аналогию между сектором и минимальной единицей адресации?**

Да

15. **Дайте определение фрагментации файла.**

Процесс, при котором файл при записи на диск разбивается на блоки различной длины, которые записываются в разные области жесткого диска.

16. **Должна ли отличаться организация доступа в программе к фрагментированному файлу от организации доступа к нефрагментированному файлу?**

Нет

17. **Почему мы акцентируем внимание на фрагментации файла, хотя абстрагируемся от внутреннего устройства оперативной памяти?**

Потому что фрагментация напрямую влияет на производительность работы с данными

18. **Дайте определение расширению файла.**

Часть названия, помогающая ОС понять формат файла

19. **Влияет ли изменение расширения файла на содержимое?**

Нет

20. **Дайте определение файловой переменной.**

Переменная типа FILE, которая используется для работы с файлами. Описана в заголовочном файле stdio.h
При открытии файла или устройства возвращается указатель на объект этого типа (файловый указатель).

21. **Дайте определение точке связывания, точке открытия, точке закрытия.**

Точка связывания – точка, где файловая переменная связывается с файлом на диске.
Точка открытия - момент, когда файл открывается для чтения или записи.
Точка закрытия - момент, когда программа закрывает. При закрытии файла все ресурсы, связанные с ним, освобождаются, и файл больше не доступен для чтения или записи.

22. **Различаются ли в Си точка связывания и точка открытия файла?**

Нет

23. **Можно ли два раза подряд закрыть один и тот же файл?**

Нет

24. **Дайте определение текстовому файлу.**

Текстовые файлы содержат, главным образом, только печатные символы. Они организованы в виде последовательности строк, каждая из которых заканчивается символом новой строки ‘\n’. В конце последней строки этот символ не является обязательным. Часто переносимый, последовательный доступ

25. **За что отвечают переменные stdin, stdout, stderr?**

stdin - стандартный поток ввода, чаще всего связан с клавиатурой, программа читает из него данные
stdout - стандартный поток вывода, чаще всего связан с дисплеем, программа выводит в него данные
stderr - стандартный поток ошибки, чаще всего связан с дисплеем, программа выводит в него возникшие ошибки

26. Как организованы режимы чтения, записи, дозаписи, псевдопроизвольного доступа?

Режим (mode)	Описание
"r"	Чтение (Read). Файл должен существовать.
"w"	Запись (Write). Если файл с таким именем не существует, он будет создан, в противном случае его содержимое будет потеряно.
"a"	Запись в конец файла (Append). Файл создаётся, если не существовал.

Функция fopen может открывать файл в текстовом или бинарном режиме. По умолчанию используется текстовый режим. Если необходимо открыть файл в бинарном режиме, то в конец строки добавляется буква b, например "rb", "wb", "ab".

r+: открывает файл для чтения и записи. Файл должен уже существовать.
w+: открывает файл для чтения и записи. Если файл уже существует, он будет перезаписан. Если файл не существует, он будет создан.
a+: открывает файл для чтения и записи, добавляя данные в конец файла. Если файл не существует, он будет создан.

27. Дайте определение бинарному файлу.

Двоичные файлы – последовательность произвольных байтов, они часто имеют сложную внутреннюю структуру. Непереносим (информация хранится, как в оперативной памяти), произвольный доступ

28. Перечислите основные макроопределения для работы с бинарными файлами.

SEEK_SET начало файла;
SEEK_CUR текущее положение файла;
SEEK_END конец файла.

29. Дайте определение типизированному файлу.

Это структура данных, в которой:
1) Все элементы «якобы» одного типа;
2) Существует «якобы» произвольный доступ (сдвиг каретки «якобы» за const время);
3) «Якобы» лежит на диске одним куском подряд.
Типизированный файл можно считать массивом на диске

30. Существует ли интерфейс в языке Си для работы с типизированными файлами?

Нет

31. **Предложите свою реализацию функций для работы с типизированным файлом целых чисел по аналогии с массивом целых чисел.**

```
// Чтение записи с текущей позиции
int read(FILE *f, int *rec);

// Запись записи в текущую позицию
int write(FILE *f, const int *rec);

// Установка "курсора" на позицию pos относительно начала файла
int set_pos(FILE *f, size_t pos);

// Установка "курсора" на позицию pos относительно origin
int set_pos_ex(FILE *f, size_t pos, int origin);

// Возвращает текущую позицию "курсора"
int get_pos(FILE *f, size_t *pos);

// Возвращает размер файла в записях
int rf_file_size(FILE *f, size_t *n_recs);

// "Обрезает" файл до текущей позиции
int rf_truncate(FILE *f);
```

32. **Какова мотивация к использованию типизированного файла перед использованием массива в памяти?**

Хранение, переносимость, сохранение между разными сеансами пользования программой.

33. **Можно ли проверять на совпадение два двоичных файла целых чисел с помощью компараторов fc, diff, cmp?**

Да

34. **В каком случае можно проверять на совпадение два двоичных файла записей? Как это связано с упаковкой и наличием строк внутри записи?**

В случае, если файлы упакованы по-разному – нельзя.
В случае наличия строк внутри записи – нельзя, так как системный компилятор проверяет побитово (сравнивается в том числе и мусор после детерминированного нуля)

35. **Расскажите об особенностях работы fread. Можно ли проверять валидность типизированного файла только с помощью функций feof и ferror?**

```
size_t fread(void *ptr, size_t size, size_t count, FILE *f);
```

Функция fread считывает из файла, связанного с файловой переменной f, данные и помещает их в буфер ptr. Количество считываемых элементов буфера указывается в переменной count, а размер каждого элемента – в переменной size. fread возвращает число удачно прочитанных элементов. Если возвращаемое значение отличается от количества элементов, значит произошла ошибка или был достигнут конец файла.

Нельзя.

36. **Расскажите о работе с внешними ресурсами по отношению к программе на примере работы с файлом. Можно ли не закрывать файл?**

Работа с внешними ресурсами, такими как файлы, часто включает в себя несколько этапов: открытие, чтение или запись данных, и закрытие.

Не закрывать файл нельзя, так как без fclose:
1) У программы может быть ограничение на количество одновременно открытых файлов.
2) В Windows, когда программа открывает файл, другие программы не могут его открыть или удалить.
3) Вывод в файл совершается не сразу, а через вспомогательный буфер. При отсутствии вызова fclose информация из него может не попасть в файл.

37. Расскажите о разделении функции на два множества относительно работы с внешними ресурсами согласно правилу Тараса Бульбы.