Вопрос 3

Определение «первого совпадающего по маске файла»:

Первый совпадающий файл определяется не только по имени, но и может зависеть от текущего признака сортировки (по имени, дате, типу или размеру). Это может быть как по умолчанию, так и задано пользователем.

**Вопрос 3.0**

**Где нарвал наврал?**

1. **Избыточные команды**: push ds и pop es могут быть лишними.
2. **Значение CX**: Установка cx в 13 может привести к ошибкам, если длина имени файла другая.
3. **Проверка данных**: Нет проверки на корректность данных в DTA.
4. **FN\_Ofs**: Убедитесь, что FN\_Ofs указывает на правильные данные.

Вопрос 4

Объяснение EQU, адреса, DTA + 1eh и offset:

EQU: Директива для определения константы.

Адрес: Это место в памяти, где хранится значение.

DTA + 1eh: Указывает на смещение от начала области DTA, на 30 байт.

Offset: Указывает на адрес начала элемента в памяти.

Вопрос 5

Содержимое регистра CX после MOV CX, -128:

После выполнения MOV CX, -128 регистр CX будет содержать значение FF80h в шестнадцатеричном виде. При использовании отрицательных значений в CX программа не будет работать корректно, так как REP ожидает положительное значение.

Вопрос 6

Текст для вывода имени файла и Handle файла:

Пример кода:

mov ah, 09h

mov dx, offset FName

int 21h

mov ax, FHandle

call PrintHandle ; Процедура для вывода значения Handle

Вопрос 7

Зачем нужны команды NOP:

Команды NOP (No Operation) могут быть использованы для выравнивания кода или для создания задержки. Если их убрать, программа может работать быстрее, но может повлиять на структуру или синхронизацию кода.

Вопрос 8

Что происходит при вызове процедуры:

При вызове процедуры управление передается по адресу, указанному в инструкции CALL, и на стек помещается адрес возврата. Это позволяет вернуться к следующей инструкции после вызова.

Вопрос 9

Разница между УБИТЬ и УДАЛИТЬ Executable:

Убить: Завершить выполнение программы, оставив её на диске.

Удалить: Убрать файл из файловой системы, уничтожив его данные.

Вопрос 10

Макрорасширение для MoveFPos:

Пример макрорасширения:

mov ax, 4200h

mov bx, F\_Handle

xor cx, cx

mov dx, FPos

int 21h

Вопрос 11

Зачем нужны команды PUSH и POP:

Эти команды используются для сохранения и восстановления значений регистров, чтобы избежать их перезаписи при выполнении других операций.

Вопрос 12

### Вопрос 12: `repe cmpsb`

#### Краткое описание

- \*\*`REPE`\*\*: Префикс, который повторяет следующую команду, пока `CX` не станет 0 и флаг нуля (ZF) установлен.

- \*\*`CMPSB`\*\*: Сравнивает байт по адресу `DS:SI` с байтом по адресу `ES:DI`. Устанавливает флаги в зависимости от результата сравнения.

#### Процесс выполнения

1. \*\*Инициализация\*\*:

- `CX` содержит количество байт.

- `SI` указывает на строку в `DS`.

- `DI` указывает на строку в `ES`.

2. \*\*Выполнение\*\*:

- `CMPSB` сравнивает байты.

- Если равны (`ZF = 1`), продолжаем.

- Если не равны или `CX = 0`, завершаем.

#### Схема

```

+--------+

| CX | --> Количество байт для сравнения

+--------+

| SI | --> Указатель на строку в DS

+--------+

| DI | --> Указатель на строку в ES

+--------+

|

+------v-------+

| CMPSB |

+--------------+

|

+------v------+

| ZF = 1? |

+------+------+

|

+--> Да: CX--, SI++, DI++, продолжаем

|

+--> Нет: Цикл завершается

```

Если нужны еще уточнения, дайте знать!Вопрос 13

Что происходит по команде RET:

При выполнении RET управление передается по адресу, находящемуся на вершине стека, и стек уменьшается.

### Вопрос 13: Что происходит по команде RET

#### Краткое описание

Команда RET выполняет следующие действия:

1. Извлекает адрес возврата из стека.
2. Устанавливает этот адрес в регистр команд (IP).
3. Увеличивает указатель стека (SP), удаляя адрес возврата.

#### Рисунок: Содержимое стека до и после RET

**До выполнения RET**:

Copy

+--------+

| SP+2 | <-- Адрес возврата (на вершине стека)

+--------+

| SP+0 | <-- Сохранённый контекст

+--------+

| SP | --> Указатель стека

+--------+

**После выполнения RET**:

Copy

+--------+

| SP | --> Теперь указывает на следующую ячейку (адрес возврата удалён)

+--------+

| (другие данные) |

+--------+

### Использование отладчика

1. **Перед RET**: Проверьте стек — адрес возврата на вершине.
2. **После RET**: Проверьте стек снова — адрес возврата удалён.

Вопрос 14

#### Сколько байт и каких записывается

При вызове процедуры KillExecutable записывается **2 байта**. Эти байты содержат команду для завершения программы.

#### Записываемые байты

* **Команда**: mov ah, 40h — это команда для записи в файл.
* **Функция**: Используется функция int 21h с ah = 40h для записи данных в файл.

#### Рисунок: Содержимое записываемых байтов

plaintext

Copy

+--------+

| 40h | <-- Значение в AH (функция записи)

+--------+

| 00h | <-- Дополнительный байт (может быть нулевым)

+--------+

### Итог

* **Количество байтов**: 2 байта.
* **Записываемые байты**: 40h и 00h.

Вопрос 15

Текст, который изменяет себя в процессе выполнения:

Пример может быть:

; Программа, которая изменяет свой вывод

mov ah, 01h ; Чтение символа

int 21h

mov [output], al

; Логика изменения...

Вопрос 16

Действия транслятора при директиве END start:

Транслятор завершает обработку модуля и определяет точку входа для выполнения программы.

Вопрос 17

Что такое «РИВЕИМИТ»:

Вероятно, это неологизм, образованный от английского "reverse engineer" (реверс-инжиниринг), указывающий на процесс анализа и восстановления исходного кода или функционала программы.