**Практическая работа 6**

1. Запись — это структура, которая объединяет несколько полей разных типов. Например, запись “Студент” может состоять из полей:

- ФИО

- Возраст

- Средний балл

2. Записи объявляются с помощью ключевого слова record. Например:

pascal

TStudent = record

Name: string;

Age: Integer;

GPA: Real;

end;

Обращение к полям записи происходит через точку: Student.Name, Student.Age.

3. Записи в виде таблицы используются для хранения структурированных данных, например, базы данных сотрудников:

|  |  |  |
| --- | --- | --- |
| ID | Имя | Зарплата |
| 1 | Ваня | 25000 |
| 2 | Маша | 30000 |

4. Конструкция with упрощает доступ к полям записи. Например:

pascal

with Student do

begin

Name := 'Иван';

Age := 20;

end;

5. Записи считываются и записываются в файл с помощью процедур Read и Write. Например:

- Открыть файл.

- Записать запись в файл.

- Закрыть файл.

6. Множество — это набор уникальных элементов одного типа. Например:

pascal

Letters = set of 'A'..'Z';

7. - Объединение: A + B.

- Пересечение: A \* B.

- Разность: A - B.

- Проверка принадлежности: if 'A' in Letters.

8. Операция IN проверяет, принадлежит ли элемент множеству. Например:

pascal

if 'A' in Letters then

WriteLn('A есть в множестве');

9. Динамические данные — это данные, память для которых выделяется во время выполнения программы. Их размер может изменяться.

10. Указатель — это переменная, хранящая адрес другой переменной. Например:

pascal

P: ^Integer;

X: Integer;

P := @X; // P указывает на X

11. Список — это структура данных, состоящая из узлов. Виды списков:

- Односвязные.

- Двусвязные.

- Кольцевые.

12. Односвязный список — это список, где каждый узел содержит данные и ссылку на следующий узел.

[Узел 1] -> [Узел 2] -> [Узел 3] -> nil

13. Алфавитно-частотный словарь хранит слова и их частоту. Основные шаги:

1. Добавление слова.

2. Поиск слова.

3. Обновление частоты.

14. Двусвязный список — это список, где каждый узел содержит ссылки на предыдущий и следующий узлы. Особенность: возможность обхода в обоих направлениях.

15. Для двусвязного списка добавляем поле Prev в структуру узла и модифицируем подпрограммы для работы с двумя ссылками.

16. Сначала отсортируем слова по частоте, затем выведем их в порядке убывания.

17. Стек — это структура данных, работающая по принципу LIFO. Пример: стопка тарелок.

18. Принцип стека: последний добавленный элемент извлекается первым.

19. Основные операции:

- Push — добавить элемент.

- Pop — извлечь элемент.

- Peek — посмотреть верхний элемент.

20. Очередь — это структура данных, работающая по принципу FIFO (First In, First Out). Пример: очередь в магазине.

21. 17 вопрос повторяется

22. Принцип очереди: первый добавленный элемент извлекается первым.

23. Основные операции:

- Enqueue — добавить элемент.

- Dequeue — извлечь элемент.

- Peek — посмотреть первый элемент.

24. Дек — это двусторонняя очередь, где элементы можно добавлять и извлекать с обоих концов. Пример: очередь с возможностью добавления и удаления с обоих концов.

25. Подпрограммы:

- AddFront — добавить в начало.

- AddRear — добавить в конец.

- RemoveFront — удалить из начала.

- RemoveRear — удалить из конца.

26. Дерево — это структура данных, состоящая из узлов. Основные элементы:

- Корень.

- Листья.

- Ветви.

27. Двоичное дерево — это дерево, где каждый узел имеет не более двух потомков.

28. Алгоритм поиска:

1. Начните с корня.

2. Сравните искомое значение с текущим узлом.

3. Перейдите в левое или правое поддерево.

29. Алгоритм поиска:

1. Если текущий узел равен искомому значению, вернуть его.

2. Если искомое значение меньше текущего узла, перейти в левое поддерево.

3. Если больше — в правое.

30. 😊