Tutorial 14 - Structures, Unions & Enums in C++

Structures in C++

- A **structure** is a user-defined data type that groups variables of different types.
- **Syntax:**

```
struct employee {
    int eId;
    char favChar;
    float salary;
};

```

- **Creating instances:**

```
employee harry;
harry.eId = 1;
harry.favChar = 'c';
harry.salary = 120000000;
cout << harry.eId << harry.favChar << harry.salary;

```

- **Using `typedef`:**

```
typedef struct employee {
    int eId;
    char favChar;
    float salary;
} ep;
ep harry;
harry.eId = 1;

```

Unions in C++

- A **union** allows multiple variables to share the same memory space.
- **Key Points:**
  - Only **one variable** can be used at a time.
  - The compiler allocates memory equal to the largest data type.
- **Syntax:**

```
union money {
    int rice;
    char car;
    float pounds;
};

```

- **Creating instances:**

```
union money m1;
m1.rice = 34;
cout << m1.rice;
```

⚠️ Accessing other fields after assigning one gives garbage values.

Enums in C++

- **Enums** are user-defined types for named constants, improving readability.
- **Syntax:**

  ```
  enum Meal { breakfast, lunch, dinner };
  Meal m1 = lunch;
  cout << m1;  // Output: 1

  ```

  Key Points:

  - Values are auto-assigned as `0, 1, 2...`.