Tutorial 57 - Virtual Functions Example + Creation Rules in C++

Virtual Functions Example

Code Snippet 1: Base Class `CWH`

```cpp
class CWH {
protected:
    string title;
    float rating;
public:
    CWH(string s, float r) : title(s), rating(r) {}
    virtual void display() {} // Virtual function
};

```

- **Base Class (`CWH`)**:
  - Contains:
    - Protected members: `title` (string) and `rating` (float).
    - Constructor to initialize `title` and `rating`.
    - A pure virtual function `display()`.

Code Snippet 2: Derived Class `CWHVideo`

```cpp
class CWHVideo : public CWH {
    float videoLength;
public:
    CWHVideo(string s, float r, float vl) : CWH(s, r), videoLength(vl) {}
    void display() {
        cout << "This is an amazing video with title " << title << endl;
        cout << "Ratings: " << rating << " out of 5 stars" << endl;
        cout << "Length of this video is: " << videoLength << " minutes" << endl;
    }
};

```

- **Derived Class (`CWHVideo`)**:
  - Adds:
    - Private member: `videoLength` (float).
    - Constructor to initialize `title`, `rating`, and `videoLength`.
    - Overrides `display()` to display video details.

Code Snippet 3: Derived Class `CWHText`

```cpp
class CWHText : public CWH {
    int words;
public:
    CWHText(string s, float r, int wc) : CWH(s, r), words(wc) {}
    void display() {
        cout << "This is an amazing text tutorial with title " << title << endl;
        cout << "Ratings of this text tutorial: " << rating << " out of 5 stars" << endl;
        cout << "No of words in this text tutorial is: " << words << " words" << endl;
    }
};

```

- **Derived Class (`CWHText`)**:
  - Adds:
    - Private member: `words` (int).
    - Constructor to initialize `title`, `rating`, and `words`.
    - Overrides `display()` to display text tutorial details.

Code Snippet 4: Main Program

```cpp
int main() {
    string title;
    float rating, vlen;
    int words;

    // For CWHVideo
    title = "Django tutorial";
    vlen = 4.56;
    rating = 4.89;
    CWHVideo djVideo(title, rating, vlen);

    // For CWHText
    title = "Django tutorial Text";
    words = 433;
    rating = 4.19;
    CWHText djText(title, rating, words);

    // Array of pointers to base class
    CWH* tuts[2];
    tuts[0] = &djVideo;
    tuts[1] = &djText;

    tuts[0]->display();
    tuts[1]->display();

    return 0;
}

```

Explanation of Key Points

1. **Base Class Virtual Function**:
   - `display()` in `CWH` is declared virtual, enabling runtime polymorphism.
2. **Derived Classes**:
   - Override `display()` to provide their specific implementations.
3. **Pointers to Base Class**:
   - `tuts` is an array of pointers to `CWH`.
   - Points to derived class objects (`djVideo` and `djText`).
4. **Function Call Behavior**:

   - When `tuts[0]->display()` or `tuts[1]->display()` is called, the appropriate derived class function executes due to the virtual function mechanism.
5. **Without `virtual`**:
   - Base class `display()` would have been called regardless of the derived class object being pointed to.

Rules for Virtual Functions

1. **Cannot be static**.
2. **Accessed via object pointers**.
3. **Can be a friend** of another class.
4. **Optional Redefinition**:
   - No need to redefine in the derived class unless required.
5. **Base Class Definition**:
   - Should be defined even if not used.

Short Notes for Notebook

Virtual Functions Example:

1. **Base Class**:
   - Contains a `virtual` function `display()` for runtime polymorphism.
   - Example: `virtual void display();`.
2. **Derived Classes**:
   - Override the `display()` function to provide specific behavior.
   - Example:

     ```
     void display() override;

     ```
3. **Pointer Array**:
   - Base class pointers (`CWH*`) can point to derived class objects.

Creation Rules:

1. Cannot be **static**.
2. Accessed via **object pointers**.
3. Redefinition in the derived class is **optional**.
4. Can be **friend functions**.

Example Output:

```
This is an amazing video with title Django tutorial
Ratings: 4.89 out of 5 stars
Length of this video is: 4.56 minutes

This is an amazing text tutorial with title Django tutorial Text
Ratings of this text tutorial: 4.19 out of 5 stars
No of words in this text tutorial is: 433 words

```