Информатика. Лабораторная работа №18

Специализированные типы данных в объектно-ориентированном программировании

Цель работы

Изучить специализированные типы данных в объектно-ориентированном программировании.

Введение

Класс является одним из основных понятий в объектно-ориентированном программировании. В классе структуры данных и функции их обработки объединяются.

Класс это тип данных, определяемый пользователем.

В классе задаются свойства и поведение какого-либо предмета или процесса в виде полей данных и функций для работы с ними.

Основными свойствами ООП являются инкапсуляция, наследование и полиморфизм.

- Объединение данных с функциями их обработки в сочетании со скрытием ненужной для использования этих данных информации называется *инкапсуляцией*
- *Наследование* — это возможность создания иерархии классов, когда потомки наследуют все свойства своих предков, могут их изменять и добавлять новые.
- полиморфизм — возможность использовать в различных классах иерархии одно имя для обозначения сходных по смыслу действий и гибко выбирать требуемое действие во время выполнения программы.

Описание классов

Класс является абстрактным типом данных, определяемым пользователем, и представляет собой модель реального объекта в виде данных и функций для работы с ними.

Данные класса называются *полями* (по аналогии с полями структуры), а функции класса — *методами*. Поля и методы называются *элементами класса*. Описание класса в первом приближении выглядит так:

class <имя>{ [private:]
<описание скрытых элементов> public:
<описание доступных элементов> };

Спецификаторы доступа private и public управляют видимостью элементов класса. Элементы, описанные после служебного слова private, видимы только внутри класса. Этот вид доступа принят в классе по умолчанию. Интерфейс класса описывается после спецификатора public.

Поля класса:

- могут иметь любой тип, кроме типа этого же класса (но могут быть указателями или ссылками на этот класс);
- могут быть описаны с модификатором const, при этом они инициализируются только один раз и не могут изменяться;
- могут быть описаны с модификатором static, но не как auto, extern и register.

Описание объектов

Конкретные переменные типа «класс» называются *экземплярами класса, или объектами.* Время жизни и видимость объектов зависит от вида и места их описания и подчиняется общим правилам C++:

```
monstr Vasia;  // Объект класса monstr с параметрами по умолчанию
monstr Super(200, 300);   // Объект с явной инициализацией
monstr stado[100];      // Массив объектов с параметрами по умолчанию
```

```
monstr *beavis = new monstr (10);   // Динамический объект
//(второй параметр задается по умолчанию)
monstr &butthead = Vasia;     // Ссылка на объект
```

Доступ к элементам объекта аналогичен доступу к полям структуры. Для этого используются операция . (точка) при обращении к элементу через имя объекта и операция -> при обращении через указатель, например:

```
int n = Vasia.get_ammo();
stado[5].draw;
```

Обратиться таким образом можно только к элементам со спецификатором public. Получить или изменить значения элементов со спецификатором private можно только через обращение к соответствующим методам.

Можно создать *константный объект,* значения полей которого изменять запрещается. К нему должны применяться только константные методы:

```
class monstr{
int get_health() const {return health;}
};
const monstr Dead(0,0);   // Константный объект
cout « Dead.get_health();
```

Константный метод:

- объявляется с ключевым словом const после списка параметров;
- не может изменять значения полей класса;
- может вызывать только константные методы;
- может вызываться для любых (не только константных) объектов.
- Рекомендуется описывать как константные те методы, которые предназначены для получения значений полей.

Отчет о работе

Отчет должен содержать:

- исходные данные задания;
- решение варианта;
- результат работы программы;
- ответы на контрольные вопросы;
- выводы.

Контрольные вопросы

- что такое класс?
- в чем состоит различие между классом и объектом?
- предположим, что классы PartTimeEmployee и FullTimeEmployee наследуют свойства класса Employee. Какими характеристиками будут обладать эти классы?
- что такое интерфейс класса?
- что такое наследование?
- что такое полиморфизм?
- что такое инкапсуляция?