

Варианты зданий

1. Графики функций
2. Арифметические и геометрические прогрессии
3. Алгебраические выражения
4. Уравнения, неравенства и их системы
5. Треугольники, четырёхугольники, многоугольники и их элементы
6. Окружность, круг и их элементы
7. Площади фигур
8. Анализ диаграмм, таблиц, графиков
9. Статистика, вероятности
10. Расчеты по формулам

Используемые элементы

1. Вход с паролем

2. Форма регистрации

3. Правила тестирования

4. 10 вопросов:

А) с переключателями

Б) с флажками

Г) с вводом ответа

5. Результаты тестирования с указанием верных и неверных ответов
(с указанием верного)

СРОКИ СДАЧИ

24.04 – ТЗ и составленный тест

22.05 – Основная часть программы + Пояснительная записка с блок
схемами

29.05 – Работающее приложение + Руководство пользователя



Объектно-ориентированное программирование в C++ Классы.

Программирование и основы
алгоритмизации



Этапы повышения абстракции

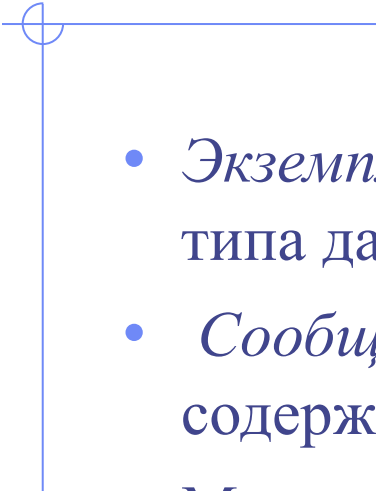
- использование функций
- описание собственных типов данных
- Объединение в модули описаний типов данных и функций, предназначенных для работы с ними

Цель

- упростить структуру программы, то есть представить ее в виде меньшего количества более крупных блоков и минимизировать связи между ними.

Класс

- *тип данных, определяемый пользователем.*
- Класс используется только через его интерфейс — детали реализации для пользователя класса несущественны.
- В классе задаются свойства и поведение какого-либо предмета или процесса в виде полей данных (аналогично структуре) и функций для работы с ними.
- В классе можно задать:
 - внутреннее представление данных в памяти компьютера,
 - множество значений, которое могут принимать величины этого типа
 - операции и функции, применяемые к этим величинам.
- Существенное свойство класса — детали его реализации скрыты от пользователей класса за интерфейсом

- 
- *Экземпляр класса или объект* — конкретные величины типа данных «класс» .
 - *Сообщение* — это запрос на выполнение действия, содержащий набор необходимых параметров.
 - Механизм сообщений реализуется с помощью вызова соответствующих функций.
 - С помощью ООП реализуется «событийно-управляемая модель», когда данные активны и управляют вызовом того или иного фрагмента программного кода

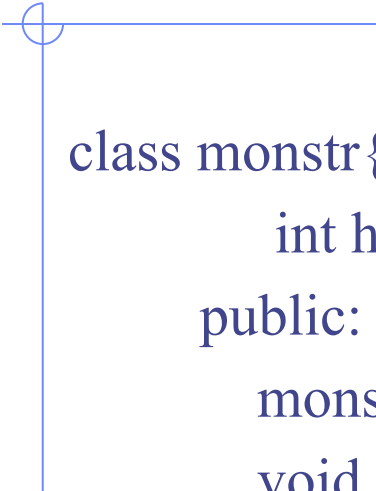
- Инкапсуляция – объединение данных с функциями их обработки в сочетании со скрыванием ненужной для использования этих данных информации
- Наследование — это возможность создания иерархии классов, когда потомки наследуют все свойства своих предков, могут их изменять и добавлять новые.
- Полиморфизм — возможность использовать в различных классах иерархии одно имя для обозначения сходных по смыслу действий и гибко выбирать требуемое действие во время выполнения программы.
- классы стандартной библиотеки `cin` и `cout`

Описание класса

- Поля – данные класса
- Методы – функции класса.
- Элементы класса – поля и методы.

Описание класса

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

```
class monstr{
    int health, ammo;
public:
    monstr(int he = 100, int am = 10){ health = he; ammo = am;};
    void draw(int x, int y, int scale, int position);
    int get_health(){return health;}
    int get_ammo(){return ammo;}
};
```

получить значения скрытых полей health и ammo из вне можно с помощью методов `get_health()` и `get_ammo()`.