Контрольные вопросы:

- ❏ (5 б.) Какие существуют способы обработки различных ошибок?
- ❏ (5 б.) В чём заключаются недостатки механизма кодов возврата?
- ❏ (5 б.) Какими особенностями обладает механизм исключений?
- ❏ (5 б.) Что такое паттерны проектирования, какова их классификация?
- ❏ (5 б.) Как формулируются гарантии безопасности исключений?

Упражнения:

- ❏ (25 б.) Продемонстрируйте использование стандартных библиотечных исключений на примере простых тестовых ситуаций.
  Указание: О библиотечных классах исключений вы можете посмотреть в этой статье. О базовом классе `std::exception` можно почитать в этой статье.
  Помните, что на диаграммах иерархии классов стрелочка указывает на базовый класс.
- ❏ (50 б.) В компьютерной многопользовательской игре “World of Phystech” персонаж игрока описывается абстрактным классом:

```
class Warrior {
private:
      // о ч к и  з д о р о в ь я  п е р с о н а ж а
      int hp;
      // у р о н ,  н а н о с и м ы й  п е р с о н а ж е м  з а  у д а р
      int damage;
public:
      // н а н е с е н и е  у д а р а  д р у г о м у  п е р с о н а ж у
      virtual void strike(Warrior& target) = 0;
};
```

Сделайте конкретные классы-потомки класса Warrior, например: Knight, Mage, Archer. Реализуйте паттерн “Фабрика” для всех классов в этой иерархии.

(25 б.) Урон класса мага (Mage) должен зависеть от выбранного магом заклинания. Реализуйте выбор заклинания при помощи паттерна “Стратегия” следующим образом.

Создайте абстрактный класс-заклинание

```
class Spell {
public:
      virtual int getDamage() = 0;
 };
```

и несколько его конкретных реализаций (Fireball, LightningBolt - на что хватит вашего воображения).

Сделайте агрегацию класса Spell в классе Mage. Функция Mage::strike() должна менять своё поведение, вызывая Spell::getDamage() выбранного заклинания.

Это творческое задание. Проявленная фантазия принесёт вам баллы выше выше номинальных. Грамотно используйте механизм исключений.