Лабораторная работа I. Вероятности.

Задача: создать систему классов, описывающую случайные величины с дискретным набором состояний; прямыми измерениями проверить статистические свойства псведослучайных величин.

Основа системы классов.

В некоторой системе для моделирования случайных величин с дискретным набором состояний определены следующие классы:

```
#include <random>

class Dice {
public:
    Dice(unsigned max, unsigned seed):
        max(max), dstr(1, max), reng(seed) { }

    unsigned roll() {
        return dstr(reng);
    }

private:
    unsigned max;
    std::uniform_int_distribution<unsigned> dstr;
    std::default_random_engine reng;
};

class ThreeDicePool {
public:
    ThreeDicePool(unsigned max,
        unsigned seed_1, unsigned seed_2, unsigned seed_3):
        max(max), dstr_1(1, max), dstr_2(1, max), dstr_3(1, max),
        reng_1(seed_1), reng_2(seed_2), reng_3(seed_3) { }

        unsigned roll() {
            return dstr_1(reng_1) + dstr_2(reng_2) + dstr_3(reng_3);
        }

private:
    unsigned max;
```

```
    std::uniform_int_distribution<unsigned> dstr_1, dstr_2, dstr_3;
    std::default_random_engine reng_1, reng_2, reng_ 3;
};
```

Класс `Dice` моделирует случайную величину с конечным набором результатов, каждый из которых равновероятен. Класс `ThreeDicePool` моделирует случайную величину, значение которой является суммой трёх случайных величин с конечным набором равновероятных результатов. Для каждого из классов написан свой вариант перегруженной функции для оценки математического ожидания:

```
double expected_value(Dice &d, unsigned number_of_rolls = 1) {
    auto accum = 0llu;
    for (unsigned cnt = 0; cnt != number_of_rolls; ++cnt)
        accum += d.roll();
    return
        static_cast<double>(accum) / static_cast<double>(number_of_rolls);
}

double expected_value(TreeDicePool &tdp, unsigned number_of_rolls = 1) {
    auto accum 0llu;
    for (unsigned cnt = 0; cnt != number_of_rolls; ++cnt)
        accum += tdp.roll();
    return
        static_cast<double>(accum) / static_cast<double>(number_of_rolls);
}
```

Задание 1. Рефакторинг. (3 балла)

Обратим внимание на две важных особенности кода: класс `ThreeDicePool` фактически использует три случайные величины, которые уже описаны классом `Dice`, код перегруженных функций `expected_value` идентичен за исключением описания входящих параметров, при этом обе функции используют идентичный для обоих классов метод `.roll`. Предложите преобразования кода, позволяющие:

- переиспользовать код класса `Dice` в классе `ThreeDicePool`;
- использовать только одну реализацию функции `expected_value` на основе полиморфизма подтипов;

- использовать класс `ThreeDicePool` для любого класса, реализующего метод `.roll`, на основе полиморфизма подтипов.

Продемонстрируйте работу классов и функций, вычислите приближённое значение математического ожидания для обеих случайных величин.

Задание 2. Штрафы и преимущества. (3 балла)

В некоторых ситуациях возникает необходимость из двух бросков выбрать наименьшее (штраф) или наибольшее (преимущество) значение. На основе составленной ранее иерархии классов опишите два дополнительных класса:

- `PenaltyDice` —класс, который позволяет для класса с методом `.roll` выполнить два броска и вернуть наименьший из двух результатов;
- `BonusDice` —класс, который позволяет для класса с методом `.roll` выполнить два броска и вернуть наибольший из двух результатов.

Для анализа работы классов `PenaltyDice` и `BonusDice` напишите функцию `double value_probability(unsigned value, Dice &d, unsigned number_of_rolls = 1)`, которая приближённо вычисляет попадание в определённое значение. Используя эту функцию постройте гистограммы вероятностей для обычного броска, броска со штрафом и броска с преимуществом для случайной величины со значениями $[1, 100]$ и для `ThreeDicePool` из трёх случайных величин со значениями $[1, 6]$.

Задание 3. Множественное наследование. (4 балла)

Используя механизм множественного наследования сконструируйте класс `DoubleDice`, который наследует одновременно и `PenaltyDice`, и `BonusDice`. Класс должен иметь конструктор с одним параметром `DoubleDice(Dice&)`, который создаёт базовые

`PenaltyDice` и `BonusDice` от одной и той же случайной величины. Вычислите математическое ожидание и постройте гистограмму вероятностей значений для `DoubleDice` сконструированного от случайной величины со значениями $[1, 100]$. Предложите реализацию `DoubleDice` без множественного наследования.