***Библиотека функций***

1. Генерация случайного числа

int generateRandomNumber(int min, int max) {

    return rand() % (max - min + 1) + min;

}

int generateNumber() {

    return rand() % 100 + 1;

}

Я объединил две данные функции, потому что они очень похожи, обе они геерируют случайное число. Различие их заключается в том, что первая принимает два аргумента – минимальное и максимальное значение диапозона из которого выбирается число, а вторая генерирует случайное число из диапозона, который указан непосредственно в теле функции.

1. Нахождение НОДа двух чисел

int nod(int a, int b) {

    while (b != 0) {

        int temp = b;

        b = a % b;

        a = temp;

    }

    return a;

}

1. Нахождение НОКа двух чисел

int nok(int a, int b) {

    return (a \* b) / nod(a, b);

}

1. Нахождение НОКа трех чисел

int nok\_3(int a, int b, int c) {

    int nok\_ab = nok(a, b);

    return nok(nok\_ab, c);

}

Эта функции может заменить два повторения функции нахождения НОКа.

1. Генерация геометрической прогрессии

void generateProgression(int progression[], int length, int commonRatio) {

    progression[0] = generateRandomNumber(1, 10);

    for (int i = 1; i < length; ++i) {

        progression[i] = progression[i - 1] \* commonRatio;

    }

}

1. Определение случайного пропущенного символа

int hideRandomNumber(int progression[], int length) {

    int index = generateRandomNumber(0, length - 1);

    int hiddenNumber = progression[index];

    progression[index] = -1;

    return hiddenNumber;

1. Вывод прогрессии с заменой пропущенного символа на двоеточие

void printProgression(int progression[], int length) {

    for (int i = 0; i < length; ++i) {

        if (progression[i] == -1) {

            cout << ".. ";

        } else {

            cout << progression[i] << " ";

        }

    }

    cout << endl;

}

***Общая логика***

В обоих программах сначала выводится первая часть текста, затем пишется условие цикла для контроля количества раундов, дальше весь код выполняется в цикле while, после окончания котоого проверяется количество правильных ответов.

cout << "Welcome to the Brain Games!" << endl;

        cout << "May I have your name? ";

        cin >> name;

        cout << "Hello, " << name << "!" << endl;

        cout << "What number is missing in the progression?" << endl;

    while(correctAnswers < 3 && rounds < 3) {

        ...

}

if(correctAnswers == 3){

        cout << "Congratulations, " << name << " , you answered all questoins correctly!" << endl;

    }else {

        cout << "The game has ended. You did not answer all questions correctly, " << name << "." << endl;

    }