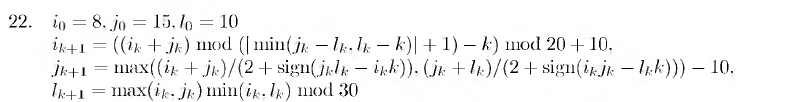
|  |  |
| --- | --- |
|  | **Отчёт по лабораторной работе** № 9 по курсу 1  Фундаментальная информатика  Студент группы M8O-101Б-23 Терентьев Михаил Андреевич, № по списку: 23  Контакты www, e-mail, icq, skype mihateren2003@gmail.com  Работа выполнена: “11 “ декабря 2023г.  Преподаватель: каф. 806 Крылов Сергей Сергеевич  Входной контроль знаний с оценкой  Отчёт сдан “13“ декабря 2023г., итоговая оценка  Подпись преподавателя |

1. **Тема**: Рекуррентные соотношения
2. **Цель работы**:составление и отладка простейшей программы на языке C итеративного характера с целочисленными рекуррентными соотношениями, задающими некоторое регулярное движение точки в целочисленном системе координат с дискретным временем и динамическим параметром
3. **Задание**: Написать программ попадания точки в треугольника   
   



1. **Оборудование** *Процессор* ***Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz 1.80 GHz*** *с ОП 8 ГБ , ТТН* ***512 ГБ*** *Встроенный монитор 13,2", FHD (1366 x 768) 16:9, частота обновления: 60 Гц*
2. *Другие устройства*
3. **Программное обеспечение:** *ПВЭМ студента* (*лабораторное*):

Программное обеспечение ПЭВМ студента, если использовалось:

Операционная система семейства Windows, наименование версия Windows 11,

Компилятор gcc. Система программирования C.

Редактор текстов Visual Studio Code версия 1.85

1. **Идея, метод, алгоритм** *решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальное описание с пред- и постусловиями):*

1. В начале программы определены константы STEPS (количество шагов) и функция funtionArea(x, y), которая возвращает true, если точка (x, y) принадлежит определенной области.

2. Затем определены функции myAbs(x), mySign(x), myMax(i, j), myMin(i, j) и myMod(x, y):

- Функция myAbs(x) возвращает абсолютное значение числа x.

- Функция mySign(x) возвращает знак числа x (-1, 0 или 1).

- Функция myMax(i, j) возвращает максимальное из двух чисел i и j.

- Функция myMin(i, j) возвращает минимальное из двух чисел i и j.

- Функция myMod(x, y) возвращает остаток от деления x на y.

3. В функции main() определяются начальные значения i, j, l и переменная step. Затем выполняется цикл while, в котором происходит вычисление новых значений i, j и l на каждом шаге:

- Сначала сохраняются текущие значения i, j и l во временные переменные tmpI, tmpJ и tmpL.

- Увеличивается счетчик шагов step.

- Проверяется, принадлежит ли текущая точка области funtionArea(i,j). Если да, то выводится номер шага и программа завершается.

- Вычисляются новые значения i, j и l согласно сложным выражениям.

4. Если после выполнения всех шагов не была достигнута цель, то выводится сообщение "В цель промах" и информация о количестве шагов и значениях i, j, l.

Таким образом, алгоритм программы вычисляет шаги для достижения цели в заданной области и выводит информацию о результатах.

1. **Сценарий выполнения работы** [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

#include <stdio.h>

#include <stdbool.h>

#include <locale.h>

const int STEPS = 50;

bool funtionArea(int x, int y){

    return (x >= -10) && (x <= 0) && (y >= x + 10) && (y <= -x + 10);

}

int myAbs(int x){

    if (x < 0){

        return -x;

    }

    return x;

}

int mySign(int x){

    if (x > 0){

        return 1;

    }

    if (x < 0){

        return -1;

    }

    return 0;

}

int myMax(int i, int j){

    if (i >= j){

        return i;

    }

    return j;

}

int myMin(int i, int j){

    if (i <= j){

        return i;

    }

    return j;

}

int myMod(int x, int y){

    return x - ((x / y) \* y);

}

int main(void)

{

    int i=-12, j=-22, l=11;

    int step=0, tmpI, tmpJ, tmpL;

    while (step <= STEPS){

        tmpI = i;

        tmpJ = j;

        tmpL = l;

        ++step;

        if (funtionArea(i,j)){

            printf("%d", step);

            return 0;

        }

        i = myMax(myMin(tmpI - tmpJ, tmpJ - tmpL) % 20, myMod(myMin(tmpI - tmpL, tmpJ - step), 20)) + 10;

        j = mySign(tmpI - tmpJ) \* myMin(myMod(tmpI, 20), myMod(tmpJ,20)) - myMod(myMax(myAbs(tmpI - tmpL), myAbs(step - 20)), 20) + 20;

        l = myMod(tmpI, 10) \* myMod(tmpJ, 10) + myMod(tmpL, 10);

    }

    setlocale(LC\_ALL, "Rus");

    printf("В цель промах\nШаг %d, %d, %d, %d", step, i, j, l);

    return 0;

}

Допущен к выполнению работы**. Подпись преподавателя**

1. **Распечатка протокола** (подклеить листинг окончательного варианта программы с текстовыми примерами, подписанный преподавателем)

#include <stdio.h>

#include <stdbool.h>

#include <locale.h>

const int STEPS = 50;

bool funtionArea(int x, int y){

    return (x >= -10) && (x <= 0) && (y >= x + 10) && (y <= -x + 10);

}

int myAbs(int x){

    if (x < 0){

        return -x;

    }

    return x;

}

int mySign(int x){

    if (x > 0){

        return 1;

    }

    if (x < 0){

        return -1;

    }

    return 0;

}

int myMax(int i, int j){

    if (i >= j){

        return i;

    }

    return j;

}

int myMin(int i, int j){

    if (i <= j){

        return i;

    }

    return j;

}

int myMod(int x, int y){

    return x - ((x / y) \* y);

}

int main(void)

{

    int i=-12, j=-22, l=11;

    int step=0, tmpI, tmpJ, tmpL;

    while (step <= STEPS){

        tmpI = i;

        tmpJ = j;

        tmpL = l;

        ++step;

        if (funtionArea(i,j)){

            printf("%d", step);

            return 0;

        }

        i = myMax(myMin(tmpI - tmpJ, tmpJ - tmpL) % 20, myMod(myMin(tmpI - tmpL, tmpJ - step), 20)) + 10;

        j = mySign(tmpI - tmpJ) \* myMin(myMod(tmpI, 20), myMod(tmpJ,20)) - myMod(myMax(myAbs(tmpI - tmpL), myAbs(step - 20)), 20) + 20;

        l = myMod(tmpI, 10) \* myMod(tmpJ, 10) + myMod(tmpL, 10);

    }

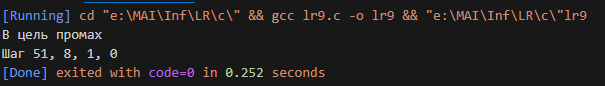
    setlocale(LC\_ALL, "Rus");

    printf("В цель промах\nШаг %d, %d, %d, %d", step, i, j, l);

    return 0;

}

*Тесты:*



1. **Дневник отладки** должен содержать дату и время сеансов отладки, и основные ошибки (ошибки в сценарии и программе, не стандартные операции) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб.  или  дом. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  |  |  |  |  |  |  |

1. **Замечание автора по существу работы**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_
2. **Выводы**

В ходе выполнения лабораторной работы я научился работе с рекуррентными соотношениями на языке Си.

Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_