Содержание

[1 Постановка задачи 2](#_Toc132474493)

[1.1 Начальная постановка задачи 2](#_Toc132474494)

[1.2 Конечная постановка задачи 2](#_Toc132474495)

[1.3 Ввод данных 2](#_Toc132474496)

[1.4 Вывод данных 2](#_Toc132474497)

[2 Метод решения 3](#_Toc132474498)

[2.1 Проверка ввода данных 3](#_Toc132474499)

[2.2 Конструкция Try … Except 3](#_Toc132474500)

[2.3 Условия ввода данных 4](#_Toc132474501)

[2.4 Общее сведения об алгоритме 4](#_Toc132474502)

[2.5 Алгоритм рекурсивного метода 4](#_Toc132474503)

[3 Описание алгоритмов решения задачи 6](#_Toc132474504)

[4 Структура данных 7](#_Toc132474505)

[5 Результаты расчетов 8](#_Toc132474506)

[Приложение А 9](#_Toc132474507)

[Приложение Б 12](#_Toc132474508)

[Приложение С 14](#_Toc132474509)

# Постановка задачи

## Начальная постановка задачи

Пользователь вводит количество крючков. Необходимо составить программу для нахождения количества последовательностей “И” и “Ш”, если:

1. Из одного крючка нельзя образовать букву;
2. Из двух крючков образуется “И”;
3. Из трех крючков образуется “Ш”;
4. Необходимо использовать все крючки.

## Конечная постановка задачи

Пользователь вводит количество крючков. Необходимо составить программу для нахождения количества последовательностей “И” и “Ш”, при условии:

1. Из одного крючка нельзя образовать букву;
2. Из двух крючков образуется “И”;
3. Из трех крючков образуется “Ш”;
4. Необходимо использовать все крючки.

Также вывести все возможные полученные последовательности.

## Ввод данных

Пользователь вводит количество крючков.

## Вывод данных

Результат выводится в следующем виде, где ResultCount − количество последовательностей из “И” и “Ш”.

# Метод решения

## Проверка ввода данных

Так как пользователь может ввести неверные данные, программа должна это отслеживать и правильно обрабатывать.

Алгоритм ввода:

IsCorrect:= False;

//Цикл ввода

While not IsCorrect do

Begin

Write(Text);

Try

//Ввод элемнта

Readln(Value);

If (Value < 2) then Writeln('Не корректный ввод')

//Если ввод корректный

Else IsCorrect:= True;

Except

//Если ввод не корректный

Writeln('Не корректный ввод');

End;

End;

В данном алгоритме используется цикл While с условием, что корректные данные ещё не введены (IsCorrect = False)

Проверка на то, что данные верны выполняется через конструкцию Try…Except и через условие в блоке Try (количество элементов последовательности должно быть больше двух и удовлетворять предельному значению номера (Описано в пункте оптимизация алгоритма)).

## Конструкция Try … Except

Конструкция Try…Except используется для отслеживания введения некорректных данных ([условия ввода данных](#_Условия_для_ввода)). В такой ситуации выполняется код в блоке Except, в который сообщает пользователю о некорректных исходных данных и просит ввести их снова.

## Условия ввода данных

Пользователь вводит количество крючков Number.

Условия ввода Number:

1. Значения могут быть только числовыми;
2. Значения могут только удовлетворять неравенству > 1;
3. Значения могут быть только целыми.

## Общее сведения об алгоритме

Для нахождения количества различных последовательностей содержащих только “И” и “Ш” использовался рекурсивный метод.

## Алгоритм рекурсивного метода

Рекурсия – это разработка алгоритма таким образом, чтобы он вызывал сам себя. Для реализации алгоритма для данной задачи используется подпрограмма AllSequences.

Данная подпрограмма получает на вход количество крючков StartCount и текущую последовательность CurrentSequence в виде “ И И Ш …”.

Данный алгоритм можно рассмотреть на примере из 8-ти крючков:

Можно заметить, что из всех позиций больше 4-х образуется две новые ветки, так как можно образовать “И” или “Ш”. Из позиции с четвёркой – одна ветка, так как можно образовать только последовательность “ИИ”. А позиции с цифрами 3 и 2 являются конечными, так как из них можно составить только по одной букве.

Таким образом если использовать для каждой точки подпрограмму и передавать в неё текущую, полученную последовательность, можно рекурсивно получать результаты для всех последовательностей:



Рисунок – Графическое представление алгоритма

После достижения конечной точки (оранжевая на рисунке), производится вывод последовательности и увеличение количества найденных последовательностей. В результате завершения рекурсии, выполняется вывод результата.

Таблица – Алгоритм работы рекурсии

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0. | AllSequences  (8,‘’,0) |  |  |  |  |
| 1. |  | AllSequences  (6,‘И’,0) |  |  |  |
| 2. |  |  | AllSequences  (4,‘ИИ’,0) | AllSequences  (2,‘ИИИ’,0) | ‘ИИИИ’ |
| 3. |  | AllSequences  (6,‘И’,1) |  |  |  |
| 4. |  |  | AllSequences  (3,‘ИШ’,1) |  | ‘ИШШ’ |
| 5. | AllSequences  (8,‘’,2) |  |  |  |  |
| 6. |  | AllSequences  (5,‘Ш’,2) |  |  |  |
| 7. |  |  | AllSequences  (3,‘ШИ’,2) |  | ‘ШИШ’ |
| 8. |  | AllSequences  (5,‘Ш’,3) |  |  |  |
| 9. |  |  | AllSequences  (2,‘ШШ’,3) |  | ‘ШШИ’ |

# Описание алгоритмов решения задачи

Таблица 2 – Описание алгоритмов решения задачи

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №  п.п. | Наименование алгоритма | Название алгоритма | Формальные  параметры | Предпола-гаемый тип реализации |
| 1 | Основной алгоритм | Вызов следующих подпрограмм:  AllSequences,  ReadInteger |  |  |
| 2 | AllSequences  (  StartCount,  CurrentSequence,  ResultCount  ) | Заданы количество крючков StartCount и текущая последовательность CurrentSequence,  подпрограмма вычисляет следующий  элемент последовательности и возвращает количество последовательностей ResultCount | StartCount – получает от фактического параметра адрес c защитой;  CurrentSequence - получает от фактического параметра адрес c защитой;  ResultCount - получает от фактического параметра адрес, возвращаемый параметр | Процедура |
| 3 | InputNum  (  Text,  Res  ) | Задана строка Text на вывод пользователю | Text – получает от фактического параметра адрес c защитой;  Res - получает от фактического параметра адрес, возвращаемый параметр | Функция. Res – возвращае-мый функцией параметр |

# Структура данных

Таблица 3 – Структура данных основной программы

|  |  |  |
| --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение |
| NumCount | Integer | Вводимое количество крючков |
| ResCount | Integer | Количество последовательностей из “И” и “Ш” |

Таблица 4 – Структура данных алгоритма AllSequences (StartCount, CurrentSequence, ResultCount)

|  |  |  |  |
| --- | --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение | Тип параметра |
| StartCount | Integer | Начальное количество крючков | Формальный |
| CurrentSequence | String | Текущая последовательность крючков | Формальный |
| ResultCount | Integer | Итоговое количество последовательностей | Формальный |

Таблица 5 – Структура данных алгоритма InputNum(Text, Res)

|  |  |  |  |
| --- | --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение | Тип параметра |
| InputText | String | Текст на вывод для пользователя | Формальный |
| IsCorrect | Boolean | Логическая переменная для проверки ввода значений | Локальный |
| Number | Integer | Вводимое значение | Локальный |
| Res | Integer | Результат – количество крючков | Формальный |

# Результаты расчетов

В ходе работы были получены следующие результаты:

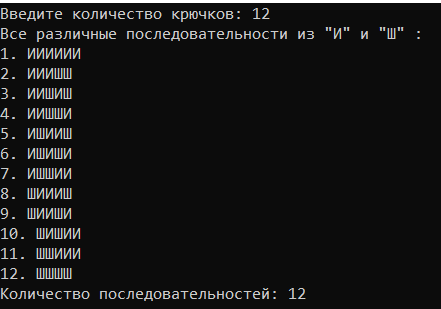


Рисунок – Результаты расчетов

Приложение А

(обязательное)

Исходный код программы с промежуточными результатами и вводом матрицы

Program Razminka3;

{

Program counts number of I and Sh that possible to create with given number of hooks. Program counts number of possibilities.

}

//Moduls that we use

uses

SysUtils;

Var

NumCount: Integer;

ResCount: Integer;

//NumCount - Number of hooks

//ResCount – Number of possibilities

//Subprogram for counting number of possibilities

{

Given recent number of hooks and recent possibility, subprogram counts next elements of line and writes the number of possibilities.

}

Procedure AllSequences(Const StartCount: integer;

Const CurrentSequence: String;

var ResultCount: Integer);

Begin

//If the number of hooks is equal or greater than 4

If StartCount >= 4 then

Begin

//Making 2 possibilities

//(one with "И", second with "Ш", more than 4)

AllSequences(StartCount - 2, CurrentSequence + 'И',

ResultCount);

if StartCount <> 4 then

AllSequences(StartCount - 3, CurrentSequence + 'Ш',

ResultCount);

End

//If 3 hooks left than there would be "Ш" in the end

Else if StartCount = 3 then

Begin

Inc(ResultCount);

Writeln(ResultCount, '. ', CurrentSequence + 'Ш')

End

//If 2, than "И"

Else

Begin

Inc(ResultCount);

Writeln(ResultCount, '. ', CurrentSequence + 'И')

End;

End;

//Subprogram for checking given line

{

User writes a string, we check it for write answer

}

Function InputNum(Const InputText: String): Integer;

Var

IsCorrect: Boolean;

Number: Integer;

//IsCorrect – Flag for checking

//Number – number of hooks

Begin

IsCorrect:= False;

//Cycle for checking given info

While not IsCorrect do

Begin

Write(InputText);

Try

//Writing number of hooks

Readln(Value);

If (Value < 2) then Writeln('Вы ввели некорректные

данные. Попробуйте снова')

//If the answer is right

Else IsCorrect:= True;

Except

//If the answer is wrong

Writeln('Вы ввели некорректные данные.

Попробуйте снова');

End;

End;

Result:= Number;

End;

Begin

//Number of hooks

NumCount:= ReadInteger('Введите количество крючков: ');

//Counting the number of sequences

Writeln('Все различные последовательности из "И" и "Ш" :');

AllSequences(NumCount, '', ResCount);

//Writing the number of sequences

Writeln('Количество последовательностей: ', ResCount);

Readln;

End.

Приложение Б

(обязательное)

Тестовые наборы для проверки вычислений

Тест 1

Исходные данные:

NumCount = 10

Ожидаемый результат :

Результирующие последовательности:

“ИИИИИ”,

“ИИШШ”,

“ИШИШ”,

“ШИИШ”,

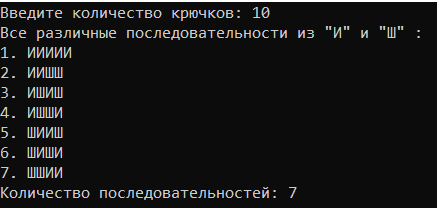
“ИШШИ”,

“ШИШИ”,

“ШШИИ”.

Количество последовательностей: 7

Полученный результат:



Тест 2

Исходные данные:

NumCount = 12

Ожидаемый результат:

Результирующие последовательности:

“ИИИИИИ”,

“ИИИШШ”,

“ИИШИШ”,

“ИШИИШ”,

“ШИИИШ”,

“ИИШШИ”,

“ИШИШИ”,

“ШИИШИ”,

“ИШШИИ”,

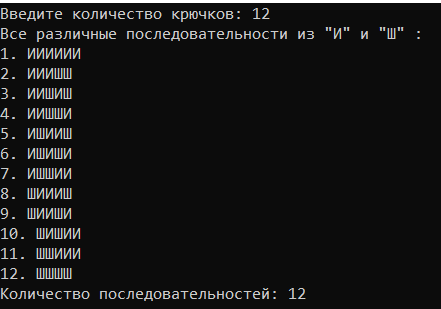
“ШИШИИ”,

“ШШИИИ”,

“ШШШШ”.

Количество последовательностей: 12

Полученный результат:



Приложение С

(обязательное)

Тестовые наборы для проверки корректности ввода

Тест 1

Исходные данные:

HookCount = 0, 1, -4, д

Ожидаемый результат: «Не корректный ввод»

Полученный результат:

