Содержание

[1 Постановка задачи 2](#_Toc132475094)

[1.1 Начальная постановка задачи 2](#_Toc132475095)

[1.2 Конечная постановка задачи 2](#_Toc132475096)

[1.3 Ввод данных 2](#_Toc132475097)

[1.4 Вывод данных 2](#_Toc132475098)

[2 Условия ввода данных 3](#_Toc132475099)

[2.1 Условия ввода данных 3](#_Toc132475100)

[2.2 Варианты ответа пользователю 3](#_Toc132475101)

[3 Метод решения 4](#_Toc132475102)

[3.1 Проверка ввода данных 4](#_Toc132475103)

[3.2 Конструкция Try … Except 4](#_Toc132475104)

[3.3 Общее сведения об алгоритме 5](#_Toc132475105)

[3.4 Алгоритм метода динамического программирования 5](#_Toc132475106)

[4 Описание алгоритмов решения задачи 6](#_Toc132475107)

[5 Структура данных 7](#_Toc132475108)

[6 Результаты расчетов 8](#_Toc132475109)

[Приложение А 9](#_Toc132475110)

[Приложение Б 12](#_Toc132475111)

[Приложение С 14](#_Toc132475112)

# 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 Value.

Условия ввода Value:

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 метод «динамического программирования».

## Алгоритм метода динамического программирования

Данный алгоритм можно рассмотреть на примере Таблицы 1:

Таблица – Зависимость количества крючков от количества последовательностей

|  |  |
| --- | --- |
| Количество крючков | Количество комбинаций |
| 1 | 0 |
| 2 | 1 |
| 3 | 1 |
| 4 | 1 |
| 5 | 2 |
| 6 | 2 |
| 7 | 3 |
| 8 | 4 |
| … | … |

Точно известно, что при 2-х крючках – ответ: 1, при 3-х – ответ: 1, при одном крючке – ответ: 0.

Можно заметить, что ответ для 4-х крючков составляется из суммы ответов при одном крючке и 2-х. Для 5-ти, из суммы для 2-х и 3-х и так далее.

Таким образом, если циклически дойти до варианта с i-тым количеством крючков, использую результаты предыдущих вычислений, можно получить результат.

# 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,  InputNum |  |  |
| 2 | AllSequences  (  StartCount,  Res  ) | Задано количество крючков StartCount,  подпрограмма вычисляет количество  последовательностей Res и возвращает его | StartCount – получает от фактического параметра адрес c защитой;  Res - получает от фактического параметра адрес, возвращаемый параметр | Функция. Res – возвращае-мый функцией параметр |
| 3 | ReadInteger  (  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 |
| ResultCount | Integer | Количество последовательностей из “И” и “Ш” |

Таблица 4 – Структура данных алгоритма GetSubsequence (HookCount, Res)

|  |  |  |  |
| --- | --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение | Тип параметра |
| StartCount | Integer | Начальное количество крючков | Формальный |
| I | Integer | Параметр цикла | Локальный |
| Num1, Num2, Num3 | Integer | Переменные хранения прошлых  количеств последовательностей | Локальный |
| CurrentCount | Integer | Текущее значение количества последовательностей | Локальный |
| Res | Integer | Итоговое количество последовательностей | Формальный |

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

|  |  |  |  |
| --- | --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение | Тип параметра |
| Text | String | Текст на вывод для пользователя | Формальный |
| IsCorrect | Boolean | Логическая переменная для проверки ввода значений | Локальный |
| Value | 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;

{

Программа высчитывает количество последовательностей из исходного количество крючков.

}

//Подключение модуля

uses

SysUtils, Windows;

//Объявление переменных

Var

NumCount: Integer;

ResultCount: Integer;

//StartHookCount - Количество крючков

//SubsequenceCount - Количество последовательностей

//Подпрограмма для вычисления

//количества последовательностей

{

Заданы текущее количество крючков, подпрограмма

вычисляет количество последовательностей.

}

Function AllSequences(Const StartCount: Integer): Integer;

Var

I: Integer;

Num1, Num2, Num3, CurrentCount: Integer;

//I - Переменная цикла

//CurCount - текущее значение количеств значений

//Count\_1, Count\_2, Count\_3 - переменные

//хранения прошлых количеств последовательностей

Begin

//Обработка базового случая

If StartCount < 4 then

CurrentCount:= 1

Else

Begin

//Присвоение начальных значений

Num1:= 0;

Num2:= 1;

Num3:= 1;

//Цикл составления количеств

//последовательностей из

//предыдущих значений

For I:= 4 to StartCount do

Begin

//Поиск текущего количества последовательностей

CurrentCount:= Num1 + Num2;

//Переприсваивание значений для

//предыдущих значений количеств

Num1:= Num2;

Num2:= Num3;

Num3:= CurrentCount;

End;

End;

Result:= CurrentCount;

End;

//Подпрограмма для ввода

//целочисленного значения

{

Задана строка, для вывода пользователю. Пользователь

вводит значение, пока не введёт корректное

}

Function InputNum(Const Text: String): Integer;

//Объявление переменных

Var

IsCorrect: Boolean;

Value: Integer;

//IsCorrect - Флаг кооректности ввода

//Value - Вводимое значение

Begin

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;

Result:= Value;

End;

Begin

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

//Ввод количества крючков

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

//Посчёт количесва возможных последовательностей

ResultCount:= AllSequences(NumCount);

//Вывод количества последовательностей

Writeln('Количество последовательностей: ', ResultCount);

Readln;

End.

Приложение Б

(обязательное)

Тестовые наборы для проверки вычислений

Тест 1

Исходные данные:

NumCount = 10

Ожидаемый результат :

Количество последовательностей: 7

Полученный результат:



Тест 2

Исходные данные:

NumCount = 12

Ожидаемый результат:

Количество последовательностей: 12

Полученный результат:



Приложение С

(обязательное)

Тестовые наборы для проверки корректности ввода

Тест 1

Исходные данные:

NumCount = 0, 1, -4, д

Ожидаемый результат: «Не корректный ввод»

Полученный результат:

