Содержание

[1 Постановка задачи 3](#_Toc116916030)

[1.1 Исходное условие задачи 3](#_Toc116916031)

[1.2 Модифицированное условие задачи 3](#_Toc116916032)

[1.3 Исходные данные 3](#_Toc116916033)

[2 Структура данных 4](#_Toc116916034)

[3 Детали и методика решения задачи 5](#_Toc116916035)

[3.1 Детали задачи 5](#_Toc116916036)

[3.2 Используемые формулы 5](#_Toc116916037)

[3.3 Проверка исходных данных с помощью цикла с постусловием 6](#_Toc116916038)

[3.4 Проверка результатов расчетов с помощью цикла с постусловием 7](#_Toc116916039)

[3.5 Использование Try … Except … End; 7](#_Toc116916040)

[4 Результаты расчетов 8](#_Toc116916041)

[Приложение А 9](#_Toc116916042)

[Приложение Б 13](#_Toc116916043)

# Постановка задачи

## Исходное условие задачи

Есть определенное количество зеленых бактерий и определенное количество тактов времени, где каждый такт времени зеленые бактерии делятся на зеленые и красные, а красные – превращаются в зеленые. Вывести общее количество бактерий через заданное количество тактов времени.

## Модифицированное условие задачи

Есть определенное количество зеленых и красных бактерий и определенное количество тактов времени, где каждый такт времени зеленые бактерии делятся на зеленые и красные, а красные – превращаются в зеленые. Вывести изначальные значения зеленых и красных бактерий, общее количество бактерий, зеленых и красных бактерий через заданное количество тактов времени.

## Исходные данные

Программа получает от пользователя количества бактерий зеленого и красного цветов, количество тактов времени.

Вывод данных происходит в следующем виде:

Количество зеленых: SumGreen

Количество красных: SumRed

После N тактов времени из Red красных и Green зеленых бактерий образовалось SumAll бактерий в сумме

Если при расчетах количество бактерий стало отрицательным (произошло переполнение типа Integer) пользователь получает следующее сообщение:

Произошло переполнение, попробуйте значения меньше

# Структура данных

Таблица 1 – Данные

|  |  |  |
| --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение |
| Red | Integer | Количество красных бактерий |
| Green | Integer | Количество зеленых бактерий |
| SumGreen | Integer | Сумма зеленых бактерий |
| SumRed | Integer | Сумма красных бактерий |
| SumGreenR | Integer | Сумма зеленых бактерий по ветке красной бактерии |
| SumGreenG | Integer | Сумма зеленых бактерий по ветке зеленой бактерии |
| SumRedR | Integer | Сумма красных бактерий по ветке красной бактерии |
| SumRedG | Integer | Сумма красных бактерий по ветке зеленой бактерии |
| SumAll | Integer | Сумма всех бактерий через N тактов времени |
| N | Integer | Количество тактов времени |
| I | Integer | Вспомогательная переменная для подсчетов |
| R1 | Real | Вспомогательная переменная для подсчетов |
| R2 | Real | Вспомогательная переменная для подсчетов |
| E | Boolean | Переменная для проверки условия на ввод неверного типа данных |
| P | Boolean | Переменная для проверки условия на случай переполнения |

# Детали и методика решения задачи

## Детали задачи

Детали задачи таковы:

* начальное значение тактов времени каждого цвета может быть только целым числом больше 0;
* значения зеленых и красных бактерий могут быть нулевыми.

## Используемые формулы

Проанализировав условия задачи, можно сделать вывод, что сумма бактерий из веток красных и зеленых бактерий изменяется как сумма чисел Фибоначчи. Так, по формуле Бине:

Кроме того, номер члена при расчете суммы бактерий по зеленой ветке будет отличаться от суммы бактерий по красной ветке, а именно:

* зеленая ветка: в сумме зеленых – N+1, в сумме красных – N;
* красная ветка: в сумме зеленых – N, в сумме красных – N-1,

где N – число тактов времени и номер члена последовательности чисел Фибоначчи; во время первого такта происходит деление бактерий. Таким образом, отличия в формуле обуславливаются тем, что:

Таблица 2 – Количество бактерий, если изначально есть 1 зеленая

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № такта | Количество зеленых бактерий | Номер члена последовательности Фибоначчи | Количество красных бактерий | Номер члена последовательности Фибоначчи |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 2 | 1 | 1 |
| 2 | 2 | 3 | 1 | 2 |
| 3 | 3 | 4 | 2 | 3 |
| 4 | 5 | 5 | 3 | 4 |
| 5 | 8 | 6 | 5 | 5 |

Таблица 3 – Количество бактерий, если изначально есть 1 красная

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № такта | Количество зеленых бактерий | Номер члена последовательности Фибоначчи | Количество красных бактерий | Номер члена последовательности Фибоначчи |
| 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 |
| 2 | 1 | 2 | 1 | 1 |
| 3 | 2 | 3 | 1 | 2 |
| 4 | 3 | 4 | 2 | 3 |
| 5 | 5 | 5 | 3 | 4 |

## Проверка исходных данных с помощью цикла с постусловием

При проверке исходных данных использовался цикл с постусловием, так как в таком случае цикл будет запущен хотя бы один раз:

Repeat

Try

Write('Введите кол-во бактерий красного и зеленого');

Writeln(' цветов соответственно: ');

Readln(Red, Green);

While (Red <= -1) or (Green <= -1) do

Begin

Write('Введите кол-во бактерий красного и зелено

го');

Writeln(' цветов соответственно: ');

Readln(Red, Green);

E := True;

End;

Writeln('Введите кол-во тактов времени (до 40): ');

Readln(N);

While (N < 0) do

Begin

Writeln('Введите кол-во тактов времени (до 40): ');

Readln(N);

E := True;

End;

E := True;

except

Writeln('Введены некорректные данные!');

Writeln('');

E := False;

End;

Until E;

С помощью этого цикла мы повторяем вывод фразы “ Введите кол-во бактерий красного и зеленого цветов соответственно: ” , повторяем получение данных от пользователя в переменные Red и Green, если было правильно введено значение, иначе фразу “Введены некорректные данные! ” до тех пор, пока полученные значения не будет выполнять всем требованиям. Так же происходит и с вводом количества тактов времени.

## Проверка результатов расчетов с помощью цикла с постусловием

При проверке результатов расчетов использовался цикл с постусловием, так как в таком случае цикл будет запущен хотя бы один раз. При проведении расчетов с большими значениями исходных данных происходит переполнение типа Integer, что приводит к отрицательным значениям. В таком случае пользователю выводится фраза «Произошло переполнение, попробуйте значения меньше» и программа запускается заново. В противном случае пользователю выводятся результаты расчетов.

## Использование Try … Except … End;

В данной задаче данные вводятся пользователем с клавиатуры. Нельзя утверждать, что пользователь введёт верные данные, а именно тип Integer.

Для этого используется «Try … Except … End;» При введении некорректных данных (это может быть как и слишком большое число, так и не число вовсе) происходит ошибка, которую считывает Except и выполняет следующие строки:

Writeln('Введены неверные данные!');

Writeln('');

E := False;

# Результаты расчетов

Вследствие выполнения программы на экран выводятся следующие результаты расчетов:

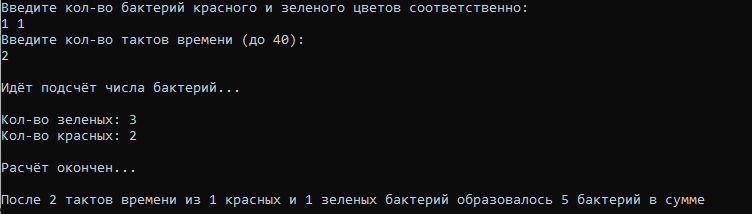


Рисунок 1– Результаты расчётов

Приложение А

(обязательное)

Исходный код программы

program Бактерии;

// Calculating sum of green and red bacteria after N

// time cycles

{$APPTYPE CONSOLE}

uses

System.SysUtils;

var Red, Green,SumGreen, SumRed, SumGreenR, SumRedR,

SumGreenG, SumRedG, SumAll, N, I : integer;

R1, R2 : real;

E, P : boolean;

// Red, Green - amount of red and green bacteria

// SumGreen, SumRed - sum of green and red bacteria

// SumGreenR, SumRedR - sum of green and red

// bacteria (for red branch)

// SumGreenG, SumRedG - sum of green and red

// bacteria (for green branch)

// SumAll - sum of green and red bacteria together

// N - amount of time cycles

// I, R1, R2 - auxiliary variables

// E - checking value

begin

// For case of overflow

Repeat

// Requesting and obtaining the initial amount of

// bacteria and time cycles

Repeat

Try

// Rerequesting and reobtaining the initial

// amount of bacteria

Write('Введите кол-во бактерий красного и ');

Writeln(' зеленого цветов соответственно: ');

Readln(Red, Green);

// if amount isn't right

While (Red <= -1) or (Green <= -1) do

Begin

Write('Введите кол-во бактерий красного и ');

Writeln(' зеленого цветов соответственно: ');

Readln(Red, Green);

// Except repeat

E := True;

End;

// Rerequesting and reobtaining the initial

// amount of time cycles

Writeln('Введите кол-во тактов времени(до 40):');

Readln(N);

// if amount isn't right

While (N < 0) do

Begin

Writeln('Введите кол-во тактов времени(до

40):');

Readln(N);

// Except repeat

E := True;

End;

// Except repeat

E := True;

// if the user gave the wrong type

except

Writeln('Введены некорректные данные!');

Writeln('');

// Go to repeat

E := False;

End;

Until E;

Writeln('');

Writeln('Идёт подсчёт числа бактерий... ');

Writeln('');

// Calculation of additional values (R1, R2)

R1 := (1+sqrt(5))/2;

R2 := (1-sqrt(5))/2;

// Raising R2 to a power of N+2

for I := 2 to N+2 do

R2 := R2 \* (1-sqrt(5))/2;

// Calculation of sum of green and red bacteria

// and sum of green and red bacteria together

// if only value of green bacteria is given

if Red = 0 then

Begin

SumGreen := round((exp((N+1)\*ln(R1)) - R2\*((1-

sqrt(5))/2)) / sqrt(5));

SumRed := round((exp((N)\*ln(R1)) - R2) /

sqrt(5));

SumAll := (SumRed + SumGreen) \* Green;

End

Else

// if only value of red bacteria is given

If Green = 0 then

Begin

SumGreen := round((exp((N)\*ln(R1)) - R2) /

sqrt(5));

SumRed := round((exp((N-1)\*ln(R1)) - R2/((1-

sqrt(5))/2)) / sqrt(5));

SumAll := (SumRed + SumGreen) \* Red;

End

Else

// if values of red and green bacteria were given

Begin

// Calculation of sum of green bacteria for

// green and red branch

SumGreenG := round((exp((N+1)\*ln(R1)) - R2\*((1-

sqrt(5))/2)) / sqrt(5));

SumGreenR := round((exp((N)\*ln(R1)) - R2) /

sqrt(5));

SumGreen := SumGreenG + SumGreenR;

// Calculation of sum of red bacteria for

// green and red branch

SumRedG := round((exp((N)\*ln(R1)) - R2) /

sqrt(5));

SumRedR := round((exp((N-1)\*ln(R1)) - R2 /((1-

sqrt(5))/2)) / sqrt(5));

SumRed := SumRedG + SumredR;

// Calculation of sum of green and red bacteria

// together for all branches

SumAll := (SumRedR+SumGreenR)\*Red + (Sum

RedG+SumGreenG)\*Green;

End;

// Values output

if (SumAll > 0) and (SumGreen > 0) and (SumRed > 0)

then

Begin

Writeln('Количество зеленых: ', SumGreen);

Writeln('Количество красных: ', SumRed);

Writeln('');

Writeln('Расчёт окончен...');

Writeln('');

Write('После ', N, ' тактов времени из ', Red);

Write(' красных и ', Green, ' зеленых бактерий');

Writeln(' образовалось ', SumAll, ' бактерий в

сумме');

// The overflow isn't here

P:= True;

End

Else

// The overflow is here

Begin

Write('Произошло переполнение, попробуйте ');

Writeln(‘значения меньше’);

// Go to repeat of all program since overflow

P:= False;

End;

Until P;

readln;

end.

Приложение Б

(обязательное)

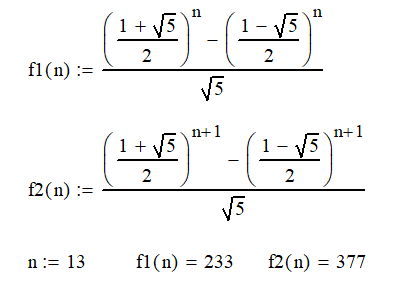
Тестовые наборы

Тест 1

Исходные данные:

* зеленые бактерии – 1;
* красные бактерии – 0;
* такты времени – 13.

Ожидаемый результат:



Полученный результат:

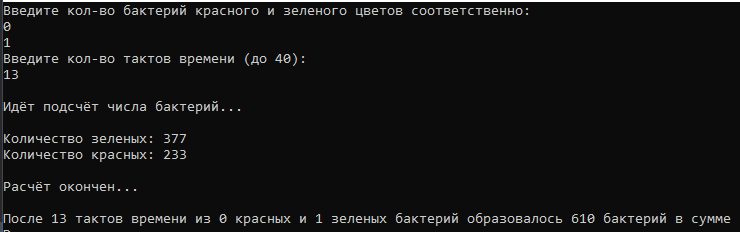


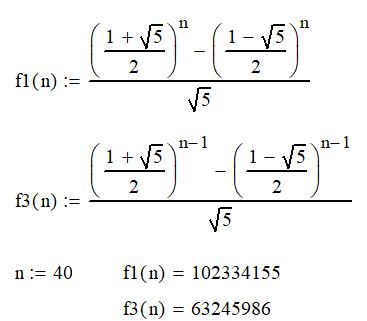
Рисунок 2 – Подсчет результатов при введении количества только зеленых бактерий

Тест 2

Исходные данные:

* зеленые бактерии – 0;
* красные бактерии – 1;
* такты времени – 40.

Ожидаемый результат:



Полученный результат:

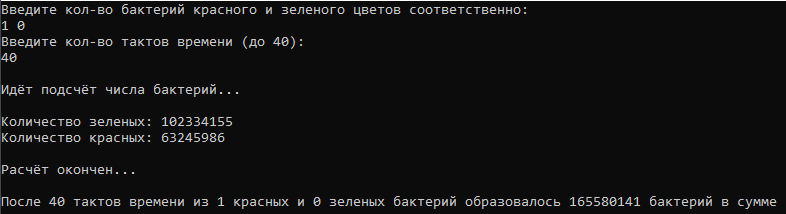


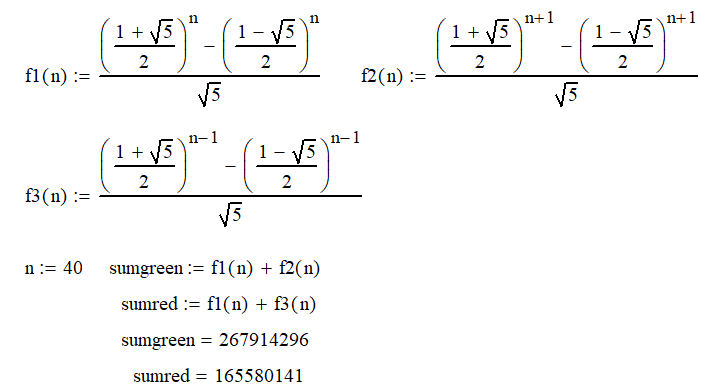
Рисунок 3 – Подсчет результатов при введении количества только красных бактерий

Тест 3

Исходные данные:

* зеленые бактерии – 1;
* красные бактерии – 1;
* такты времени – 40.

Ожидаемый результат:



Полученный результат:

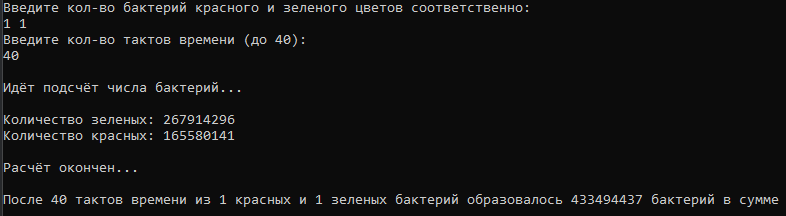


Рисунок 4 – Подсчет результатов при введении количества зеленых и красных бактерий

Тест 4

Таблица 5 – Тестовые ситуации

|  |  |  |
| --- | --- | --- |
| Тест | Тестовые ситуации (вместе с расчётами) | Проверка полученного результата |
| 1. |  | [См. Детали задачи](#_Детали_задачи) |
| 2. |  | [См. Детали задачи](#_Детали_задачи) |
| 3. |  | [См. Детали задачи](#_Детали_задачи) |