Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования (ОАиП)

ОТЧЕТ

по разминке №1

Тема работы: Задача про монахов

Выполнил

студент: гр. 251003 Панкратьев Е.С.

Проверил: Фадеева Е.П.

Минск 2022

**СОДЕРЖАНИЕ**

[1 Исходная постановка задачи 3](#_Toc116664552)

[2 Усложненная постановка задачи 4](#_Toc116664553)

[3 Методика решения 5](#_Toc116664554)

[3.1 Кратое описание алгоритма решения задачи 5](#_Toc116664555)

[3.2 Оптимизация 5](#_Toc116664556)

[3.3 Условия выполнения программы 5](#_Toc116664557)

[3.4 Преимущества и недостатки 5](#_Toc116664558)

[4 Структура данных 6](#_Toc116664559)

[5 Результаты расчетов 7](#_Toc116664560)

[Приложение А 8](#_Toc116664561)

[Приложение Б 11](#_Toc116664562)

# Исходная постановка задачи

Имеется 3 группы монахов: ведущие, простые и ученики. 100 монахов имели 100 пирогов. Монахи должны есть пироги по числу: ведущие по 10, простые по 5, ученики по 0.5. Программа должна вычислить и вывести на экран комбинацию групп монахов для того, чтобы все пироги были съедены.

# Усложненная постановка задачи

Имеется 3 группы монахов: ведущие, простые и ученики. Вводится общее количество монахов и пирогов. Также количество пирогов, по которому едят монахи из ведущей группы, простой группы и группы учеников. Программа должна вычислить и вывести на экран комбинацию групп монахов для того, чтобы все пироги были съедены.

# Методика решения

## Кратое описание алгоритма решения задачи

Во втором варианте решения (используя два цикла) перебираем ведущих монахов, а затем простых. Для нахождения учеников, от общего количества монахов отнимем ведущих и простых.

Далее считаем общее количество съеденных пирогов при данной комбинации монахов. Если оно равно общему количеству пирогов, тогда комбинация монахов найдена.

## Оптимизация

Для того, чтобы сократить количество итераций, найдем максимальное количество ведущих монахов, которое может быть. Для этого возьмём целую часть от деления всех пирогов на пироги, по которому едят монахи из ведущий группы. Ведущих монахов перебираем от 0 до максимального количества ведущих монахов. Если количество всех монахов больше, чем максимальное количество, тогда максимальному количеству ведущих монахов присваиваем количество всех монахов. Простых монахов перебираем от 0 до количества всех монахов минус количества ведущих монахов в итерации.

## Условия выполнения программы

Условия ввода:

1. Общее количество пирогов и монахов, а также количество пирогов, по которому едят монахи из ведущей группы, простой группы и группы учеников должно быть больше или равно нулю.
2. Общее количество монахов должно быть целочисленным.

## Преимущества и недостатки

Преимуществом данного варианта является допустимость ввода одинакового количество съедаемых пирогов для простых и учеников.

Недостатком данного варианта является большое количество итераций, относительного первого варианта.

# Структура данных

Таблица 1 – Данные

|  |  |  |
| --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение |
| MaxM1 | Integer | Максимальное количество ведущих монахов |
| M1 | Integer | Количество ведущих монахов |
| M2 | Integer | Количество простых монахов |
| M3 | Integer | Количество учеников-монахов |
| K1 | Real | Количество пирогов, по которому едят монахи из ведущий группы |
| K2 | Real | Количество пирогов, по которому едят монахи из простой группы |
| K3 | Real | Количество пирогов, по которому едят монахи из группы учеников |
| M | Integer | Общее количество монахов. |
| Pies | Real | Общее количество пирогов. |
| CyclePies | Real | Общее количество съеденных пирогов |
| Comb | Boolean | Индикатор существования комбинации |

# Результаты расчетов

При вводе значений из исходной формулировки задачи, на экран выводятся следующие результаты расчетов:

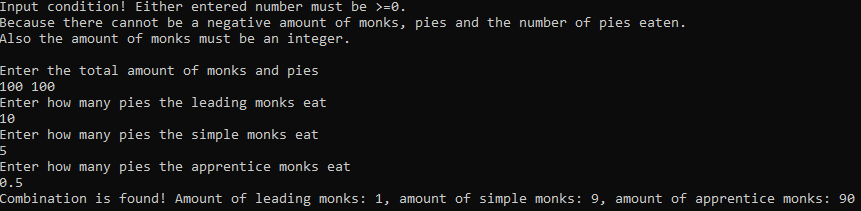


Рисунок 1 – Результаты расчетов

Приложение А

(обязательное)

Исходный код программы

Program Project2;

{

There are 3 groups of monks: leading, simple and apprentice.

Introduced how many monks and how many pies.

Also introduced how much each monk eats.

The program must calculate the combination of monks.

}

{$APPTYPE CONSOLE}

Var

MaxM1, M1, M2, M3, M: integer;

K1, K2, K3, CyclePies, Pies: real;

Comb: boolean;

//MaxM1 - maximum number of leading monks

//M1 - number of leading monks

//M2 - number of simple monks

//M3 - number of apprentice monks

//K1 - the amount, which the leading monks eats

//K2 - the amount, which the simple monks eats

//K3 - the amount, which apprentice simple monks eats

//M - total amount of monks

//Pies - total amount of pies

//CyclePies - amount of pies in a cycle

//Comb - if the condition is incorrect, then there

//will be an error.

Begin

Writeln('Input condition! Either entered number must be >=0.');

Writeln('Because there cannot be a negative amount of monks, pies and the amount of pies eaten.');

Writeln('Also the amount of monks must be an integer.');

Writeln;

Writeln('Enter the total amount of monks and pies');

Read(M);

Readln(Pies);

Writeln('Enter how many pies the leading monks eat');

Readln(K1);

Writeln('Enter how many pies the simple monks eat');

Readln(K2);

Writeln('Enter how many pies the apprentice monks eat');

Readln(K3);

if (M<0) or (Pies<0) or (K1<0) or (K2<0) or (K3<0) then

Writeln('Input condition violated! Restart the program.')

else

begin

//{Check K1 to find the maximum amount. If K1=0, then the maximum amount is also 0.

if K1 = 0 then

MaxM1:=0

//Else we will find the maximum number using

//the formula

else

begin

//Find the integer maximum number of leaders.

MaxM1:= Trunc(Pies/K1);

//The maximum amount of leading monks must be

///less than the total amount of monks.

///Else the maximum number is equated to the total //number

if MaxM1 > M then

MaxM1:= M;

end;

//If the program finds a combination, the Comb will //be true.

Comb:= False;

//Iterate over leading monks, varying from 0 to the ///maximum amount of leading monks.

For M1:=0 to MaxM1 do

begin

//Iterate over simple monks, varying from 0 to

//the total number of monks minus amount of leading

//(because amount of apprentice should be >0)

For M2:=0 to (M-M1) do

begin

///Find apprentice monks

M3:= M - M1 - M2;

//Find the amount of all pies in this iterate.

CyclePies:= M1\*K1 + M2\*K2 + M3\*K3;

//If this amount equals entered amount,

//combination is found.

if (CyclePies = Pies) then

begin

Comb:= True;

Writeln('Сombination is found! Amount of leading monks: ', M1, ', amount of simple monks: ', M2, ', amount of apprentice monks: ', M3);

end;

end;

end;

//If the combination is not found, write an error

if Comb = False then

Writeln('Error. There is no such combination of monks.');

end;

Readln;

End.

Приложение Б

(обязательное)

Тестовые наборы

Тест 1

Исходные данные:

126 монахов, 400 пирогов.

Лидер ест 3 пирога, простой – 3, ученик – 10.

Первый вариант:

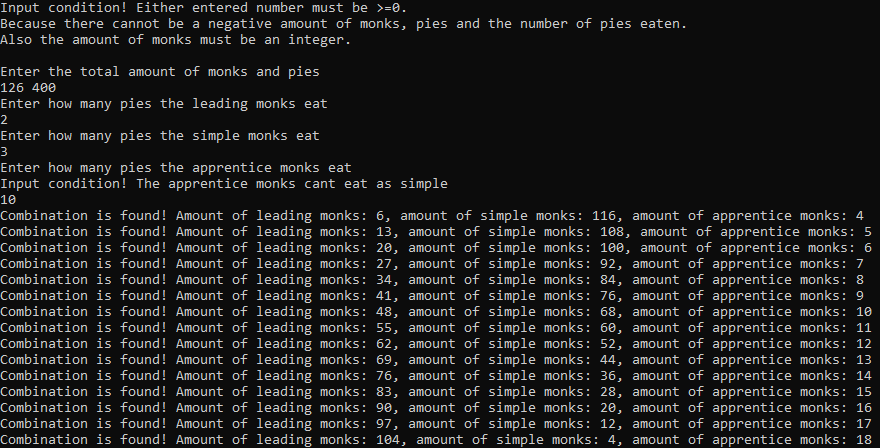


Рисунок 2 – Результаты расчетов

Второй вариант:

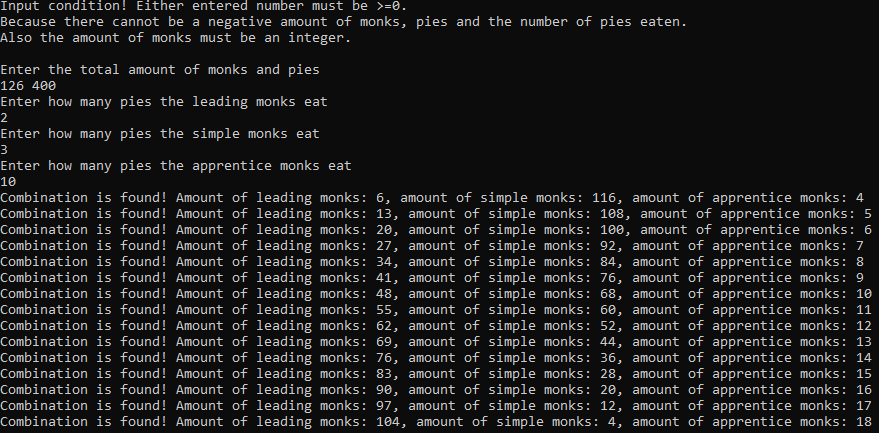


Рисунок 3 – Результаты расчетов

Тест 2

Исходные данные:

300 монахов, 463 пирогов.

Лидер ест 1 пирога, простой – 30, ученик – 8.

Первый вариант:

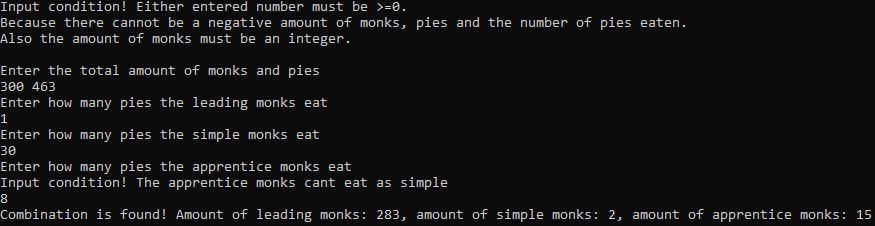


Рисунок 4 – Результаты расчетов

Второй вариант:

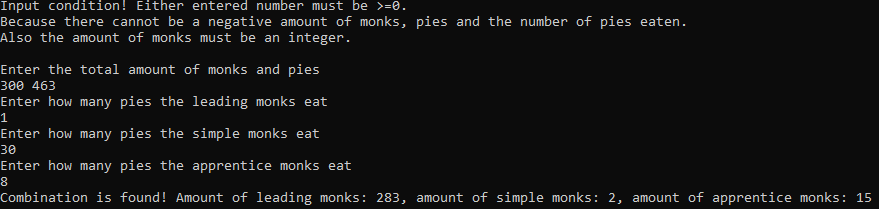


Рисунок 5 – Результаты расчетов

Тест 3

Исходные данные:

500 монахов, 600 пирогов.

Лидер ест 30 пирога, простой – 40, ученик – 50.

Первый вариант:

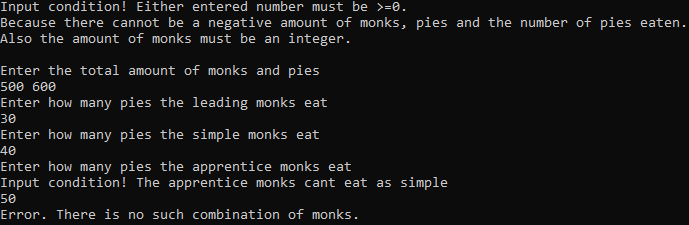


Рисунок 6 – Результаты расчетов

Второй вариант:

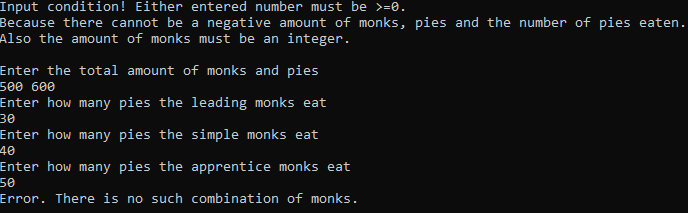


Рисунок 7 – Результаты расчетов

Тест 4

Исходные данные:

40 монахов, 100 пирогов.

Лидер ест 2 пирога, простой – 4, ученик – 4.

Первый вариант:

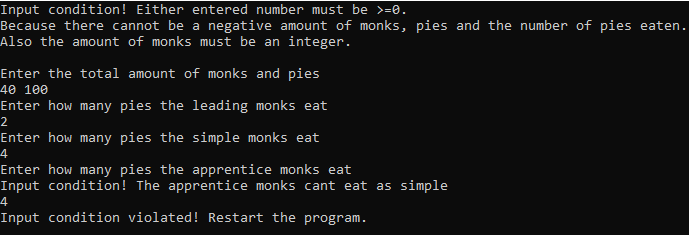


Рисунок 8 – Результаты расчетов

Второй вариант:

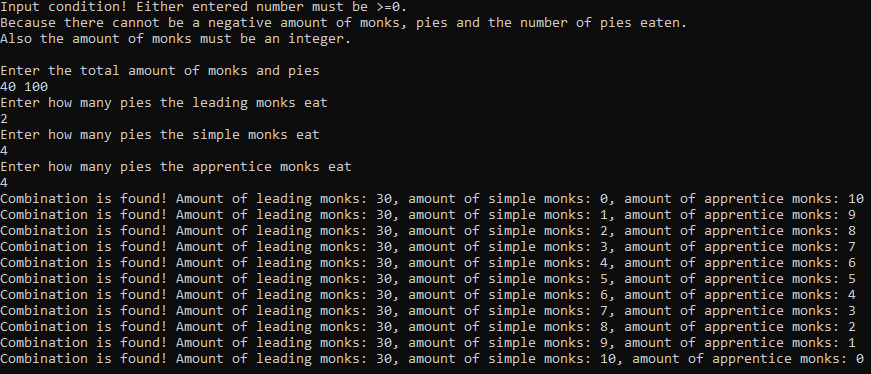


Рисунок 9 – Результаты расчетов

Тест 5

Исходные данные:

200 монахов, -500 пирогов.

Лидер ест 2 пирога, простой – 4, ученик – 10.

Первый вариант:

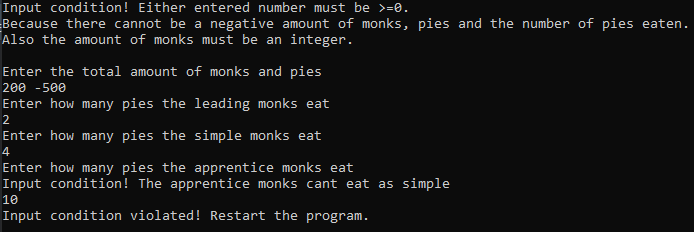


Рисунок 10 – Результаты расчетов

Второй вариант:

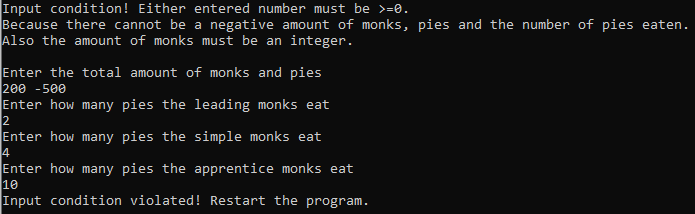


Рисунок 11 – Результаты расчетов