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 лабораторной работе № 6

Тема работы: Пользовательские типы данных

Выполнил

студент: гр. 151003 Барановский Р.А.

Проверил: Фадеева Е.П.

Минск 2021

содержание

[1 Постановка задачи 3](#_Toc87632342)

[2 Текстовый алгоритм решения задачи 4](#_Toc87632343)

[3 Структура данных 6](#_Toc87632344)

[4 Схема алгоритма решения задачи по Гост 19.701-90 7](#_Toc87632345)

[5 Результаты расчетов 11](#_Toc87632346)

[Приложение А 12](#_Toc87632347)

[Приложение Б 16](#_Toc87632348)

# Постановка задачи

Дан временной период(число, месяц, год – число, месяц, год). Вычислить количество пятниц, приходящихся на 13 число.

# Текстовый алгоритм решения задачи

Таблица – Алгоритм решения

|  |  |
| --- | --- |
| Номер  шага | Назначение шага |
| 1 | Ввод startDay |
| 2 | Ввод startMonth |
| 3 | Ввод startYear |
| 4 | Ввод finishDay |
| 5 | Ввод finishMonth |
| 6 | Ввод finishYear |
| 7 | Проверка выполнения условия startMonth < 3. Если условие истинно, то идти к шагу 8, иначе к шагу 11 |
| 8 | startMonthTransformed := startMonth + 12 |
| 9 | startYearTransformed := startYear - 1 |
| 10 | Конец условия условия startMonth < 3. Идти к шагу 13 |
| 11 | startMonthTransformed := startMonth |
| 12 | startYearTransformed := startYear |
| 13 | j := startYearTransformed div 100 |
| 14 | k := startYearTransformed mod 100 |
| 15 | wd := (startDay + (13\*(startMonthTransformed + 1) div 5) + k + (k div 4) + (j div 4) - 2\*j) mod 7 |
| 16 | currentWeekDay := Week\_Day(wd) |
| 17 | currentMonth := Month(startMonth - 1) |
| 18 | currentDay := Day(startDay) |
| 19 | currentYear := startYear |
| 20 | Начало цикла А1. Проверка выполнения условия currentYear <= finishYear. Если условие истинно, то идти к шагу 21, иначе к шагу 97 |
| 21 | Проверка выполнения условия (currentYear mod 4 = 0) and not((currentYear mod 100 = 0) and (currentYear mod 400 <> 0). Если условие истинно, то идти к шагу 22, иначе к шагу 24 |
| 22 | leapYear := true |
| 23 | Конец условия (currentYear mod 4 = 0) and not((currentYear mod 100 = 0) and (currentYear mod 400 <> 0). Идти к шагу 25 |
| 24 | leapYear := false |
| 25 | Проверка выполнения условия currentYear = finishYear. Если условие истинно, то идти к шагу 26, иначе к шагу 67 |
| 26 | Начало цикла А2. Проверка выполнения условия currentMonth <= Month(finishMonth - 1). Если условие истинно, то идти к шагу 27, иначе к шагу 65 |

Продолжение таблицы 1

|  |  |
| --- | --- |
| 27 | Проверка выполнения условия currentMonth = Month(finishMonth – 1). Если условие истинно, то идти к шагу 28, иначе к шагу 40 |
| 28 | Начало цикла А3. Проверка выполнения условия currentDay <= Day(finishDay). Если условие истинно, то идти к шагу 29, иначе к шагу 38 |
| 29 | Проверка выполнения условия (currentDay = 13) and (currentWeekDay = Пятница). Если условие истинно, то идти к шагу 30, иначе к шагу 32 |
| 30 | inc(Friday13) |
| 31 | Вывод MonthArr[currentMonth], currentYear |
| 32 | currentDay := succ(currentDay) |
| 33 | Проверка выполнения условия currentWeekDay = пятница. Если условие истинно, то идти к шагу 34, иначе к шагу 36 |
| 34 | currentWeekDay := Week\_Day(0) |
| 35 | Конец условия currentWeekDay = пятница. Идти к шагу 37 |
| 36 | currentWeekDay := succ(currentWeekDay) |
| 37 | Конец цикла А3. Идти к шагу 28 |
| 38 | currentDay := Day(1) |
| 39 | Конец условия currentMonth = Month(finishMonth – 1). Идти к шагу 63 |
| 40 | Проверка выполнения условия leapYear = false. Если условие истинно, то идти к шагу 41, иначе к шагу 52 |
| 41 | Начало цикла А4. Проверка выполнения условия currentDay <= Day(NotLeapDays[currentMonth]). Если условие истинно, то идти к шагу 42, иначе к шагу 51 |
| 42 | Проверка выполнения условия (currentDay = 13) and (currentWeekDay = Пятница). Если условие истинно, то идти к шагу 43, иначе к шагу 45 |
| 43 | inc(Friday13) |
| 44 | Вывод MonthArr[currentMonth], currentYear |
| 45 | currentDay := succ(currentDay) |
| 46 | Проверка выполнения условия currentWeekDay = пятница. Если условие истинно, то идти к шагу 47, иначе к шагу 49 |
| 47 | currentWeekDay := Week\_Day(0) |
| 48 | Конец условия currentWeekDay = пятница. Идти к шагу 50 |
| 49 | currentWeekDay := succ(currentWeekDay) |
| 50 | Конец цикла А4. Идти к шагу 41 |
| 51 | Конец условия leapYear = false. Идти к шагу 62 |

Продолжение таблицы 1

|  |  |
| --- | --- |
| 52 | Начало цикла А5. Проверка выполнения условия currentDay <= Day(LeapDays[currentMonth]). Если условие истинно, то идти к шагу 53, иначе к шагу 62 |
| 53 | Проверка выполнения условия (currentDay = 13) and (currentWeekDay = Пятница). Если условие истинно, то идти к шагу 54, иначе к шагу 56 |
| 54 | inc(Friday13) |
| 55 | Вывод MonthArr[currentMonth], currentYear |
| 56 | currentDay := succ(currentDay) |
| 57 | Проверка выполнения условия currentWeekDay = пятница. Если условие истинно, то идти к шагу 58, иначе к шагу 60 |
| 58 | currentWeekDay := Week\_Day(0) |
| 59 | Конец условия currentWeekDay = пятница. Идти к шагу 61 |
| 60 | currentWeekDay := succ(currentWeekDay) |
| 61 | Конец цикла А5. Идти к шагу 52 |
| 62 | currentDay := Day(1) |
| 63 | currentMonth := succ(currentMonth) |
| 64 | Конец цикла А2. Идти к шагу 26 |
| 65 | currentMonth := Month(0) |
| 66 | Конец условия currentYear = finishYear. Идти к шагу 95 |
| 67 | Начало цикла А6. Проверка выполнения условия currentMonth <= Month(11). Если условие истинно, то идти к шагу 68, иначе к шагу 94 |
| 68 | Проверка выполнения условия leapYear = false. Если условие истинно, то идти к шагу 69, иначе к шагу 81 |
| 69 | Начало цикла А7. Проверка выполнения условия currentDay <= Day(NotLeapDays[currentMonth]). Если условие истинно, то идти к шагу 70, иначе к шагу 79 |
| 70 | Проверка выполнения условия (currentDay = 13) and (currentWeekDay = Пятница). Если условие истинно, то идти к шагу 71, иначе к шагу 73 |
| 71 | inc(Friday13) |
| 72 | Вывод MonthArr[currentMonth], currentYear |
| 73 | currentDay := succ(currentDay) |
| 74 | Проверка выполнения условия currentWeekDay = пятница. Если условие истинно, то идти к шагу 75, иначе к шагу 77 |
| 75 | currentWeekDay := Week\_Day(0) |

Продолжение таблицы 1

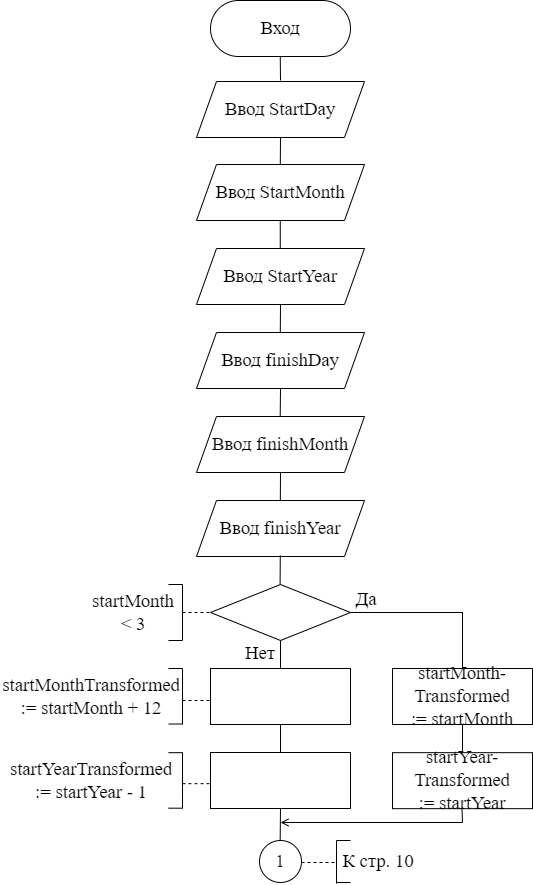
|  |  |
| --- | --- |
| 76 | Конец условия currentWeekDay = пятница. Идти к шагу 78 |
| 77 | currentWeekDay := succ(currentWeekDay) |
| 78 | Конец цикла А7. Идти к шагу 69 |
| 79 | currentDay := Day(1) |
| 80 | Конец условия leapYear = false. Идти к шагу 92 |
| 81 | Начало цикла А8. Проверка выполнения условия currentDay <= Day(LeapDays[currentMonth]). Если условие истинно, то идти к шагу 82, иначе к шагу 91 |
| 82 | Проверка выполнения условия (currentDay = 13) and (currentWeekDay = Пятница). Если условие истинно, то к шагу 83, иначе к шагу 85 |
| 83 | inc(Friday13) |
| 84 | Вывод MonthArr[currentMonth], currentYear |
| 85 | currentDay := succ(currentDay) |
| 86 | Проверка выполнения условия currentWeekDay = пятница. Если условие истинно, то идти к шагу 87, иначе к шагу 89 |
| 87 | currentWeekDay := Week\_Day(0) |
| 88 | Конец условия currentWeekDay = пятница. Идти к шагу 90 |
| 89 | currentWeekDay := succ(currentWeekDay) |
| 90 | Конец цикла А8. Идти к шагу 81 |
| 91 | currentDay := Day(1) |
| 92 | currentMonth := succ(currentMonth) |
| 93 | Конец цикла А6. Идти к шагу 67 |
| 94 | currentMonth := Month(0) |
| 95 | inc(currentYear) |
| 96 | Конец цикла А1. Идти к шагу 20 |
| 97 | Вывод Friday13 |
| 98 | Останов. |

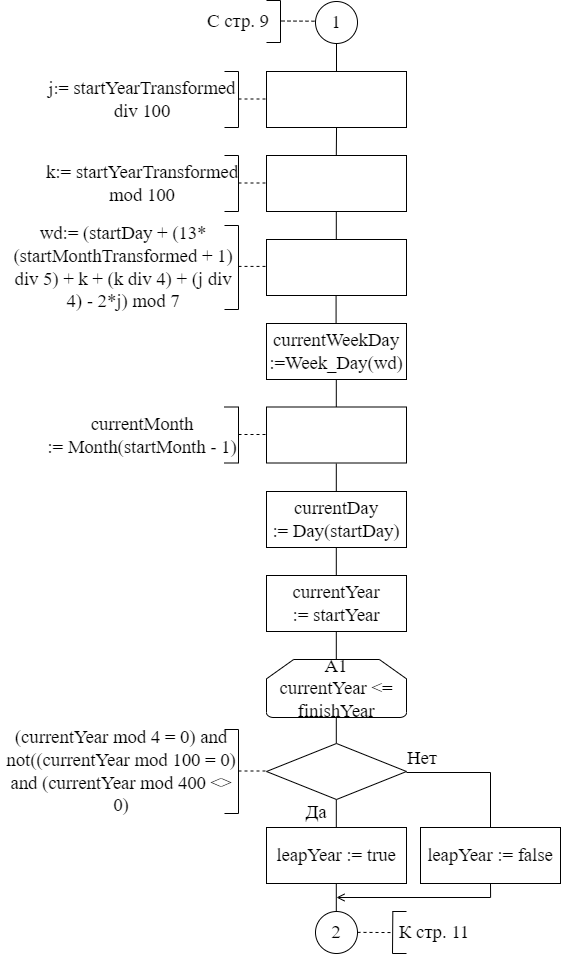
# Структура данных

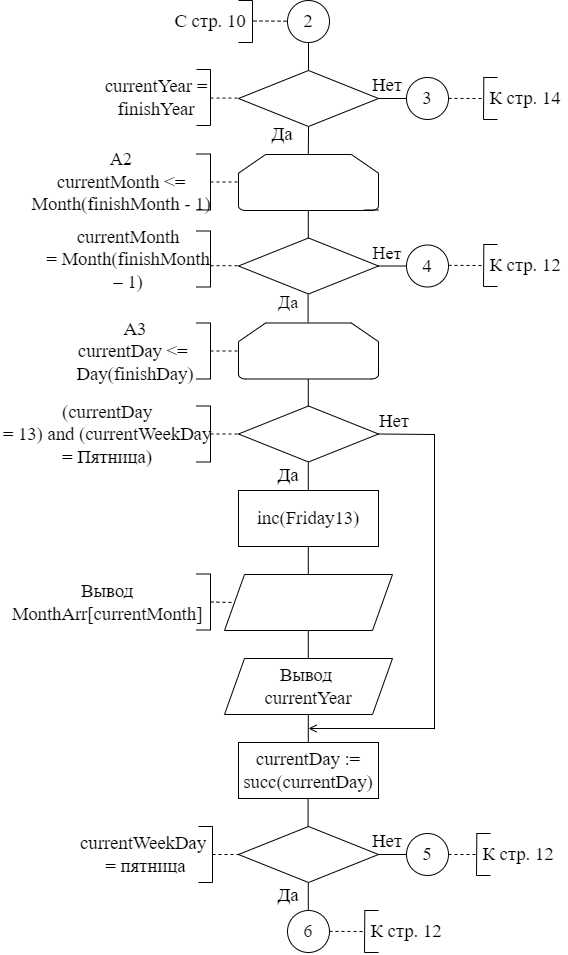
Таблица - Данные

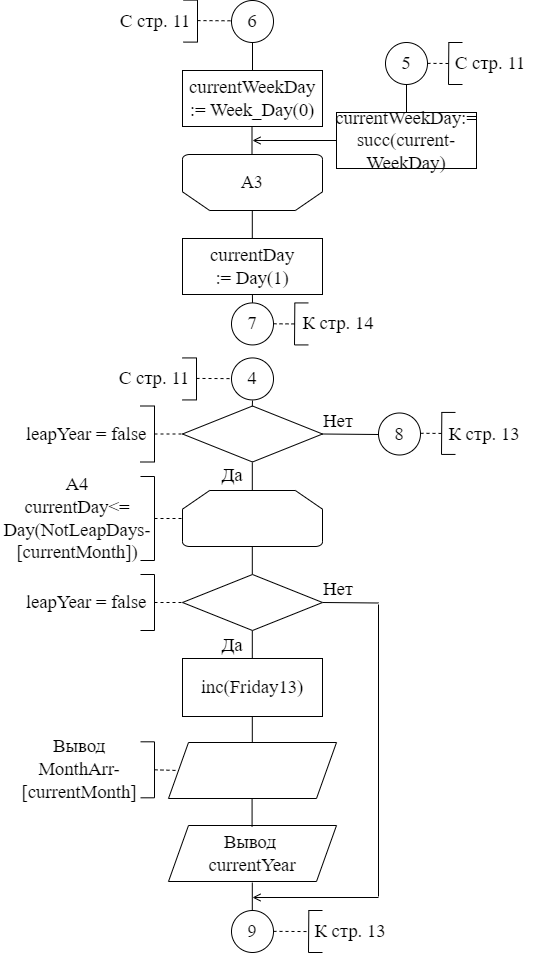
|  |  |  |
| --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение |
| NotLeapDays | Array of byte | Массив, элементами которого являются количество дней в месяцах невисокосного года |
| LeapDays | Array of byte | Массив, элементами которого являются количество дней в месяцах високосного года |
| MonthArr | Array of string | Массив, содержащий названия месяцев |
| currentWeekDay | Пользовательский тип данных:  «Перечисление» | Текущий день недели в цикле = (Суббота, Воскресенье, Понедельник, Вторник, Среда, Четверг, Пятница) |
| currentMonth | Пользовательский тип данных:  «Перечисление» | Текущий месяц в цикле = (Январь, Февраль, Март, Апрель, Май, Июнь, Июль, Август, Сентябрь, Октябрь, Ноябрь, Декабрь) |
| currentDay | Пользовательский тип данных:  «Диапазон» | Текущий день в цикле = 1..31 |
| startDay | Integer | Начальный день |
| startMonth | Integer | Начальный месяц |
| startYear | Integer | Начальный год |
| finishDay | Integer | Конечный день |
| finishMonth | Integer | Конечный месяц |
| finishYear | Integer | Конечный год |
| currentYear | Integer | Текущий год в цикле |
| wd | Integer | Индекс дня недели начальной даты |
| startMonthTransformed | Integer | Преобразованный для формулы Зеллера начальный месяц |
| startYearTransformed | Integer | Преобразованный для формулы Зеллера начальный год |
| j | Integer | Часть формулы Зеллера |
| k | Integer | Часть формулы Зеллера |
| Friday13 | Integer | Количество пятниц 13 |
| leapYear | Boolean | Принимает значение true, если проверяемый в цикле год високосный, false в противном случае |

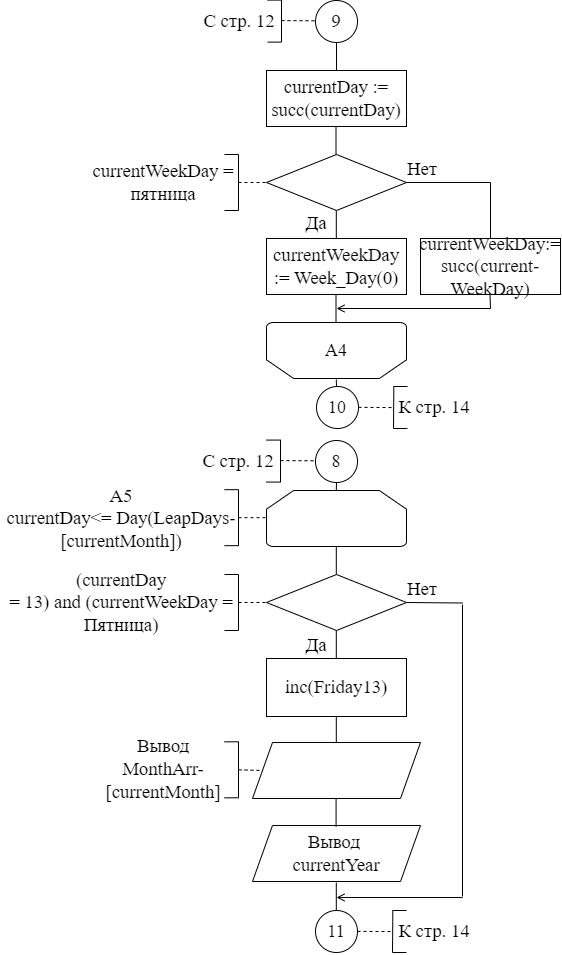
# Схема алгоритма решения задачи по Гост 19.701-90

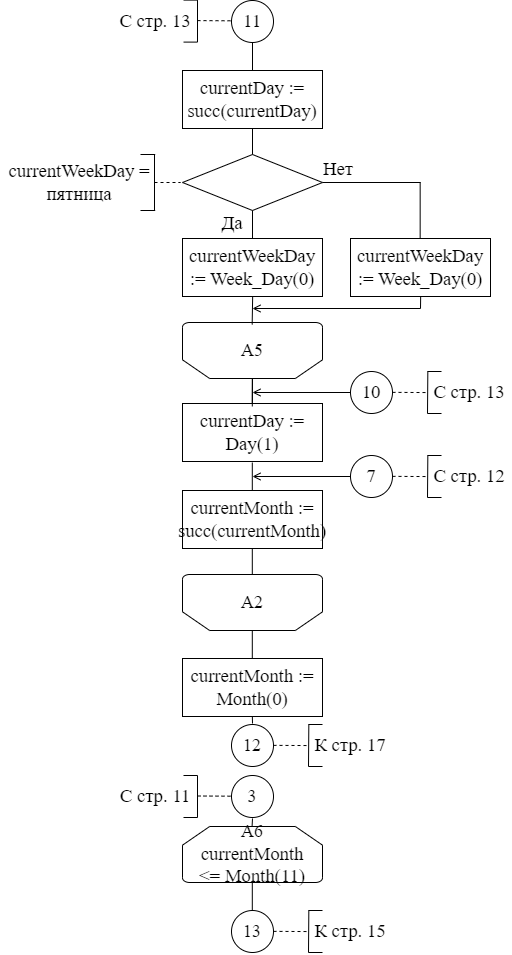


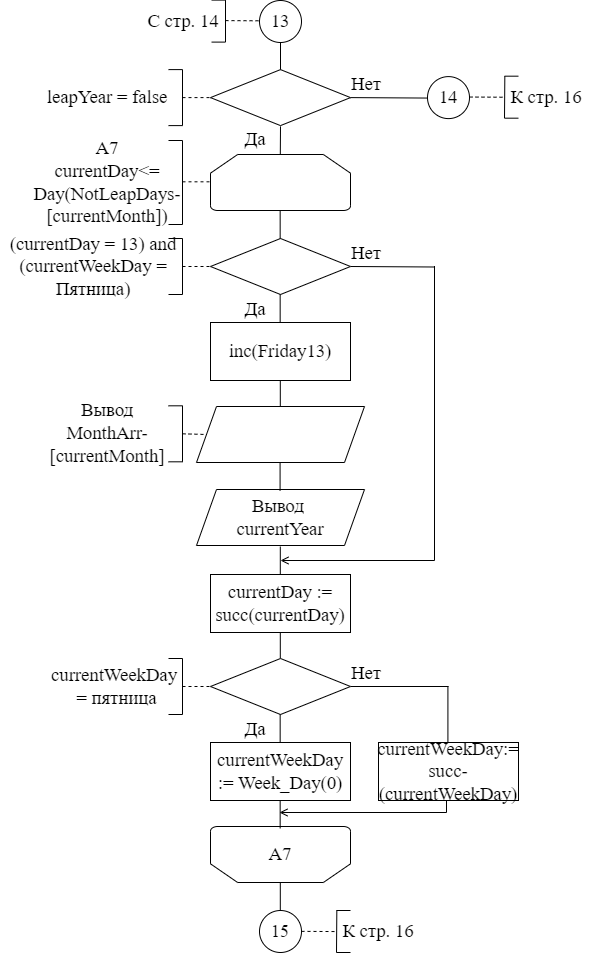


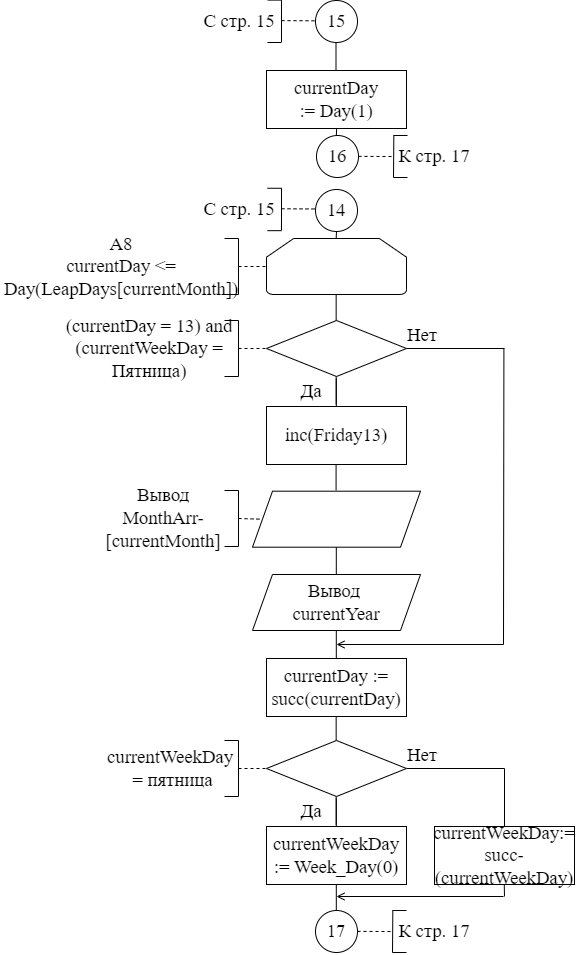












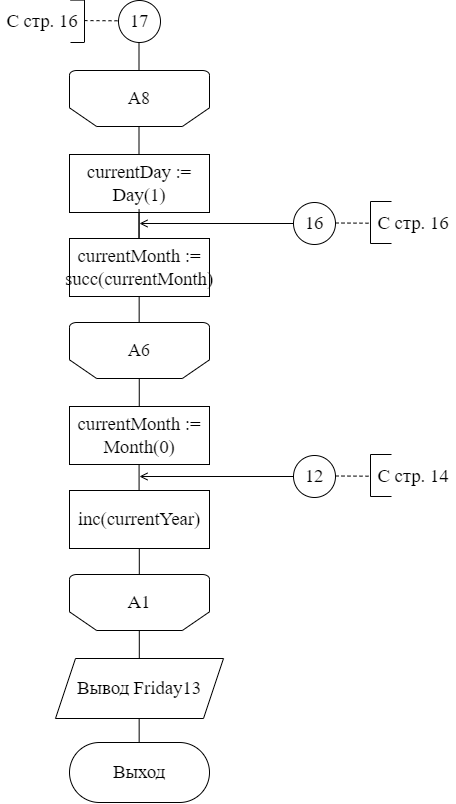


Рисунок 1 - Схема алгоритма решения задачи по ГОСТ 19.701-90

# Результаты расчетов

Вследствие результатов программы на экран выводятся следующие результаты расчетов:

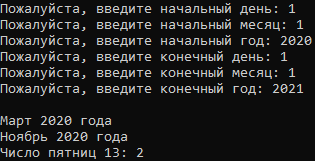


Рисунок – результаты расчетов

Приложение А

(обязательное)

Исходный код программы

Program Dates;

//Program calculates how many Friday 13-th were in certain //period of time

//Console app

{$APPTYPE CONSOLE}

//Modules declaration

Uses

System.SysUtils;

//Types declaration

Type

Month = (Январь, Февраль, Март, Апрель, Май, Июнь, Июль,

Август, Сентябрь, Октябрь, Ноябрь, Декабрь);

Week\_Day = (Суббота, Воскресенье, Понедельник, Вторник,

Среда, Четверг, Пятница);

Day = 1..31;

{ Month - months type

Week\_Day - days of week type

Day - days type }

//Constants declaration

Const

NotLeapDays:array[Month] of Byte = (31, 28, 31, 30, 31,

30, 31, 31, 30, 31,

30, 31);

LeapDays:array[Month] of Byte = (31, 29, 31, 30, 31, 30,

31, 31, 30, 31, 30, 31);

MonthArr: array[Month] of String = ('Январь', 'Февраль',

'Март', 'Апрель',

'Май', 'Июнь',

'Июль', 'Август',

'Сентябрь', 'Октябрь',

'Ноябрь', 'Декабрь');

{ NotLeapDays - days in months of not leap year

LeapDays - days in months of leap year

MonthArr - contains strings of months in order to output

them if necessary }

//Variables declaration

Var

currentWeekDay: Week\_Day;

currentMonth: Month;

currentDay: Day;

error: integer;

startDay, startMonth, startYear, finishDay, finishMonth,

finishYear, currentYear: integer;

wd, startMonthTransformed, startYearTransformed, j,

k: integer;

Friday13: integer;

startDayStr, startMonthStr, startYearStr, finishDayStr,

finishMonthStr, finishYearStr: String;

checker, leapYear: boolean;

{ currentWeekDay - current day of week in cycle

currentMonth - current month in cycle

currentDay - current date in cycle

error - variable which is <> 0 if user made input mistake

and 1 otherwise

startDay - start day

startMonth - start month

startYear - start year

finishDay - finish day

finishMonth - finish month

finishYear - finish year

currentYear - current year in cycle

wd - index of week day of the start day, 0 - saturday,

1 - sunday, etc.

startMonthTransformed - transformed month in order to

use it in Zellers formula

startYearTransformed - transformed year in order to

use it in Zellers formula

j - part of the formula which calculates index of current

day of week

k - part of the formula which calculates index of current

day of week

Friday13 - contains number of fridays 13-th in inputted

period

startDayStr - mistakes checker for startDay

startMonthStr - mistakes checker for startMonth

startYearStr - mistakes checker for startYear

finishDayStr - mistakes checker for finishDay

finishMonthStr - mistakes checker for finishMonth

finishYearStr - mistakes checker for finishYear

checker - variable which is used to prevent mistakes

because of leap years

leapYear - true if year is leap, false otherwise }

//Start the program

Begin

//Input start day

repeat

write('Пожалуйста, введите начальный день: ');

readln(startDayStr);

val(startDayStr, startDay, error);

if (error <> 0) or ((startDay < 1) or (startDay > 31))

then

writeln('Ошибка ввода');

until (error = 0) and (startDay >= 1) and

(startDay <= 31);

//Input start month

repeat

write('Пожалуйста, введите начальный месяц: ');

readln(startMonthStr);

val(startMonthStr, startMonth, error);

if (startDay > 29) and (startMonth = 2) then

checker := false

else

checker := true;

if (error <> 0) or ((startMonth < 1) or

(startMonth > 12)) or (checker = false) then

writeln('Ошибка ввода');

until (error = 0) and (startMonth >= 1) and

(startMonth <= 12) and (checker = true);

//Input start Year

repeat

write('Пожалуйста, введите начальный год: ');

readln(startYearStr);

val(startYearStr, startYear, error);

if (startYear mod 4 = 0) and not((startYear mod 100

= 0) and (startYear mod 400 <> 0)) then

leapYear := true

else

leapYear := false;

if (leapYear = false) and (startMonth = 2) and

(startDay > 28) then

checker := false

else

checker := true;

if (error <> 0) or (startYear < 1) or (checker = false) then

writeln('Ошибка ввода');

until (error = 0) and (startYear >= 1) and

(checker = true);

//Input finish day

repeat

write('Пожалуйста, введите конечный день: ');

readln(finishDayStr);

val(finishDayStr, finishDay, error);

if (error <> 0) or ((finishDay < 1) or

(finishDay > 31)) then

writeln('Ошибка ввода');

until (error = 0) and (finishDay >= 1) and

(finishDay <= 31);

//Input finish month

repeat

write('Пожалуйста, введите конечный месяц: ');

readln(finishMonthStr);

val(finishMonthStr, finishMonth, error);

if (finishDay > 29) and (finishMonth = 2) then

checker := false

else

checker := true;

if (error <> 0) or ((finishMonth < 1) or

(finishMonth > 12)) or (checker = false) then

writeln('Ошибка ввода');

until (error = 0) and (finishMonth >= 1) and

(finishMonth <= 12) and (checker = true);

//Input finish year

repeat

write('Пожалуйста, введите конечный год: ');

readln(finishYearStr);

val(finishYearStr, finishYear, error);

if (finishYear mod 4 = 0) and not((startYear mod 100

= 0) and (startYear mod 400 <> 0)) then

leapYear := true

else

leapYear := false;

if (leapYear = false) and (finishMonth = 2) and

(finishDay > 28) then

checker := false

else

checker := true;

if (error <> 0) or (finishYear < 1) or

(checker = false) or ((finishYear < startYear)

or ((finishMonth < startMonth) and (finishYear <=

startYear)) or ((finishDay <= startDay) and

(finishMonth <= startMonth) and

(finishYear <= startYear))) then

writeln('Ошибка ввода');

until (error = 0) and (finishYear >= 1) and

(checker = true) and ((finishYear > startYear) or

((finishMonth > startMonth) and

(finishYear >= startYear)) or

((finishDay > startDay) and

(finishMonth >= startMonth) and

(finishYear >= startYear)));

//Space between input and output

writeln;

//Transform start month and start year in order to insert

//them into Zellers formula

if startMonth < 3 then

begin

startMonthTransformed := startMonth + 12;

startYearTransformed := startYear - 1;

end

else

begin

startMonthTransformed := startMonth;

startYearTransformed := startYear;

end;

//Find 1-st part of Zellers formula

j := startYearTransformed div 100;

//Find 2-nd part Zellers formula

k := startYearTransformed mod 100;

//Insert all components to Zellers formula and get index

//of week day of start date as a result

wd := (startDay + (13\*(startMonthTransformed + 1) div 5)

+ k + (k div 4) + (j div 4) - 2\*j) mod 7;

//Set currentWeekDay to the original value

currentWeekDay := Week\_Day(wd);

//Set currentMonth to the original value

currentMonth := Month(startMonth - 1);

//Set currentDay to the original value

currentDay := Day(startDay);

//Set currentYear to the original value

currentYear := startYear;

//Calculations

while (currentYear <= finishYear) do

begin

//Leap year check

if (currentYear mod 4 = 0) and not((currentYear mod 100

= 0) and (currentYear mod 400 <> 0)) then

leapYear := true

else

leapYear := false;

//If current year equals to finish year

if currentYear = finishYear then

begin

//While current month less or equals to finishmonth

while currentMonth <= Month(finishMonth - 1) do

begin

//If current month equals to finish month

if currentMonth = Month(finishMonth - 1) then

begin

//While current day less or equals to finish day

while currentDay <= Day(finishDay) do

begin

//If we met friday 13-th

if (currentDay = 13) and  
 (currentWeekDay = Пятница) then

begin

inc(Friday13);

writeln(MonthArr[currentMonth],' ',

currentYear, ' года');

end;

//Increment current day

currentDay := succ(currentDay);

//Increment current day of week

if currentWeekDay = пятница then

currentWeekDay := Week\_Day(0)

else

currentWeekDay := succ(currentWeekDay);

end;

//Set current day to the first day of next month

currentDay := Day(1);

end

//If current month doesn't equals to finish month

else

begin

//If year is not leap

if leapYear = false then

begin

//While current day less or equals to number of

//days in current month of not leap year

while currentDay <=

Day(NotLeapDays[currentMonth]) do

begin

//If we met friday 13-th

if (currentDay = 13) and

(currentWeekDay = Пятница) then

begin

inc(Friday13);

writeln(MonthArr[currentMonth],' ',

currentYear, ' года');

end;

//Increment current day

currentDay := succ(currentDay);

//Increment current day of week

if currentWeekDay = пятница then

currentWeekDay := Week\_Day(0)

else

currentWeekDay := succ(currentWeekDay);

end;

end

//If year is leap

else

begin

//While current day less or equals to number of

//days in current month of leap year

while currentDay <= Day(LeapDays[currentMonth])

do

begin

//If we met friday 13-th

if (currentDay = 13) and

(currentWeekDay = Пятница) then

begin

inc(Friday13);

writeln(MonthArr[currentMonth],' ',

currentYear, ' года');

end;

//Increment current day

currentDay := succ(currentDay);

//Increment current day of week

if currentWeekDay = пятница then

currentWeekDay := Week\_Day(0)

else

currentWeekDay := succ(currentWeekDay);

end;

end;

//Set current day to the first day of next month

currentDay := Day(1);

end;

//Increment current month

currentMonth := succ(currentMonth);

end;

//When current year is over

currentMonth := Month(0);

end

//If current year doesn't equals to finish year

else

begin

//While current month less of equals to december

while currentMonth <= Month(11) do

begin

//If year is not leap

if leapYear = false then

begin

//While current day less or equals to number of

//days in current month of not leap year

while currentDay <=

Day(NotLeapDays[currentMonth]) do

begin

//If we met friday 13-th

if (currentDay = 13) and

(currentWeekDay = Пятница) then

begin

inc(Friday13);

writeln(MonthArr[currentMonth],' ',

currentYear, ' года');

end;

//Increment current day

currentDay := succ(currentDay);

//Increment current day of week

if currentWeekDay = пятница then

currentWeekDay := Week\_Day(0)

else

currentWeekDay := succ(currentWeekDay);

end;

//Set current day to the first day of next month

currentDay := Day(1);

end

//If year is leap

else

begin

//While current day less or equals to number of

//days in current month of leap year

while currentDay <= Day(LeapDays[currentMonth])

do

begin

//If we met friday 13-th

if (currentDay = 13) and

(currentWeekDay = Пятница) then

begin

inc(Friday13);

writeln(MonthArr[currentMonth],' ',

currentYear, ' года');

end;

//Increment current day

currentDay := succ(currentDay);

//Increment current day of week

if currentWeekDay = пятница then

currentWeekDay := Week\_Day(0)

else

currentWeekDay := succ(currentWeekDay);

end;

//Set current day to the first day of next month

currentDay := Day(1);

end;

//Increment current month

currentMonth := succ(currentMonth);

end;

//When current year is over

currentMonth := Month(0);

end;

//Increment current year

inc(currentYear);

end;

//Output number of fridays 13-th

write('Число пятниц 13: ');

writeln(Friday13);

//Final operations

readln;

End.

Приложение Б

(обязательное)

Тестовые наборы(Расчеты)

Тест 1

Тестовая ситуация: Стандартные даты

Исходные данные: startDay = 08

startMonth = 12

startYear = 2005

finishDay = 22

finishMonth = 06

finishYear = 2017

Ожидаемый результат:

Январь 2006 года

Октябрь 2006 года

Апрель 2007 года

Июль 2007 года

Июнь 2008 года

Февраль 2009 года

Март 2009 года

Ноябрь 2009 года

Август 2010 года

Май 2011 года

Январь 2012 года

Апрель 2012 года

Июль 2012 года

Сентябрь 2013 года

Декабрь 2013 года

Июнь 2014 года

Февраль 2015 года

Март 2015 года

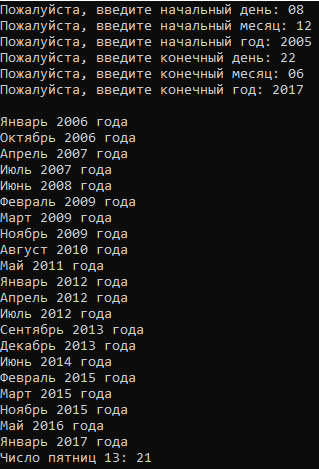
Ноябрь 2015 года

Май 2016 года

Январь 2017 года

Число пятниц 13: 21

Полученный результат:



Тест 2

Тестовая ситуация: На выбранный период не выпадают пятницы 13

Исходные данные: startDay = 15

startMonth = 12

startYear = 2021

finishDay = 31

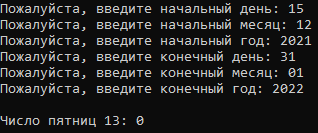
finishMonth = 01

finishYear = 2022

Ожидаемый результат:

Число пятниц 13: 0

Полученный результат:



Тест 3

Тестовая ситуация: Начальная дата – 29 февраля(год - високосный)

Исходные данные: startDay = 29

startMonth = 02

startYear = 2000

finishDay = 15

finishMonth = 08

finishYear = 2005

Ожидаемый результат:

Октябрь 2000 года

Апрель 2001 года

Июль 2001 года

Сентябрь 2002 года

Декабрь 2002 года

Июнь 2003 года

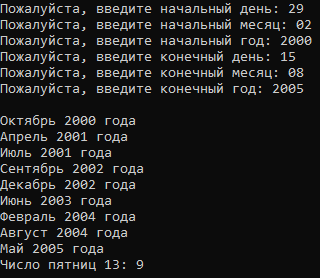
Февраль 2004 года

Август 2004 года

Май 2005 года

Число пятниц 13: 9

Полученный результат:



Тест 4

Тестовая ситуация: Конечная дата – 29 февраля(год - високосный)

Исходные данные: startDay = 07

startMonth = 11

startYear = 1973

finishDay = 29

finishMonth = 02

finishYear = 1976

Ожидаемый результат:

Сентябрь 1974 года

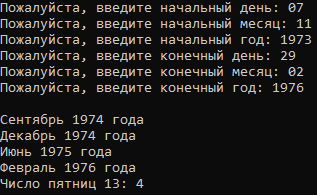
Декабрь 1974 года

Июнь 1975 года

Февраль 1976 года

Число пятниц 13: 4

Полученный результат:



Тестовые наборы(Проверка ввода)

Тест 5

Тестовая ситуация: Конечная дата раньше начальной

Исходные данные: startDay = 01

startMonth = 06

startYear = 2019

finishDay = 15

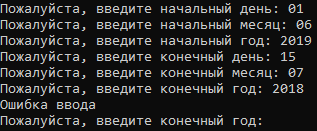
finishMonth = 07

finishYear = 2018

Ожидаемый результат:

Ошибка ввода

Полученный результат:



Тест 6

Тестовая ситуация: Ввод 29 февраля во время невисокосного года

Исходные данные: startDay = 29

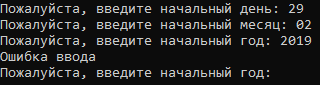
startMonth = 02

startYear = 2019

Ожидаемый результат:

Ошибка ввода

Полученный результат:



Тест 7

Тестовая ситуация: Ввод букв в поле для чисел

Исходные данные: startDay = ‘f’

Ожидаемый результат:

Ошибка ввода

Полученный результат:

