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 разминочной работе №2

Тема работы: Шифрование

Выполнил

студент: гр. 551003 Дементей В.С.

Проверила: Фадеева Е.П.

Минск 2016

Содержание

[1 Постановка задачи 3](#_Toc447913055)

[2 Описание алгоритмов 4](#_Toc447913056)

[3 Результаты расчетов и тестирование программы 6](#_Toc447913057)

[3.1 Тест 1 6](#_Toc447913058)

[3.2 Тест 2 6](#_Toc447913059)

[Приложение А 7](#_Toc447913060)

# Постановка задачи

Зашифровать и расшифровать введенных текст. Для шифрования текста производить его запись в квадратную матрицу и начиная с центрального элемента матрицы раскрывать по спирали в случайном направлении и записывать результат раскрытия в строку.

Шифрование:

Ввод: Текст для шифрования.

Вывод: Зашифрованный текст.

# Описание алгоритмов

Таблица – Описание алгоритмов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Наименование  алгоритма | Назначение  Алгоритма | Формальные  параметры | Рекомендуемый  тип реализации |
| 1 | Основной  алгоритм | Ввод шифруемой строки. Вызов алгоритмов:  Createarray  (entestring,first, lengthAr,NArray)  Createstr  (CArray, lengthStrInAr, totalstr)  Cypher  (CArray,sim,  lengthAr, firststatus, totalstr, direction)  Uncpher  (CArray,lengthAr,  sim, totalstr, status,  direction) |  |  |
| 2 | Createarray  (entestring,  first, lengthAr, NArray) | Преобразование введенной строки entestring в квадратную матрицу NArray размера lengthAr | entestring,  first, lengthAr, NArray  Возвращаемые параметры:  NArray, lengthAr | Процедура |
| 3 | Createstr  (CArray, lengthStrInAr, totalstr) | Преобразование матрицы CArray в строку totalstr | CArray, lengthStrInAr, totalstr  Возвращаемые параметры:  totalstr | Процедура |
| 4 | Cypher  (CArray, sim,  lengthAr, firststatus, totalstr,  direction) | Шифрование текста в матрице CArray и занесение результата шифрования в  Totalstr | CArray, sim,  lengthAr, firststatus, totalstr,  direction  Возвращаемые параметры:  Totalstr, firststatus, direction | Процедура |
| 5 | Uncpher  (CArray,  lengthAr,sim, totalstr, status,  direction) | Раcшифровка текста в матрице CArray и занесение результата шифрования в  Totalstr | CArray,  lengthAr,sim, totalstr, status,  direction  Возвращаемые параметры:  Totalstr | Процедура |

# Результаты расчетов и тестирование программы

## Тест 1

Исходные данные: “Hello,world!”

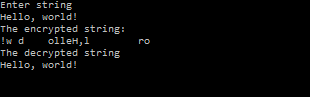
Ожидаемый результат:

Enter string:

Hello,world!

The encrypted string:

!w d olleH,l ro

The decrypted string:

Hello,world

Полученный результат:

## Тест 2

Исходные данные:

Если жизнь тебя обманет,

Не печалься, не сердись!

В день уныния смирись:

День веселья, верь, настанет.

Сердце в будущем живёт;

Настоящее уныло:

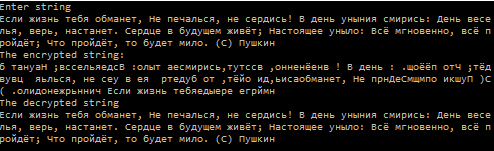
Всё мгновенно, всё пройдёт;

Что пройдёт, то будет мило.

Пушкин

Ожидаемый результат: Зашифрованная строка

Полученный результат:



Приложение А

(обязательное)

Исходный код программы

**Program** Cipher;

{$APPTYPE CONSOLE}

**Uses**

SysUtils,

math;

**Type**

TArD=**array of array of** char;

TStat = 1..4;

**Var**

lengthAr:integer;

frontstring:string;

NArray,totalArray:TArD;

totalstr:string;

firststatus:TStat;

direction:integer;

{Создание и заполнение массива значениями из строки}

**procedure** createarray(enterstring:string;first:boolean;

**var** NArray:TArD;**var** lengthAr:integer);

**var**

i,j,border:integer;

**begin**

lengthAr:=ceil(sqrt(length(enterstring)));

**if** (lengthAr **mod** 2)= 0 **then**

lengthAr:=lengthAr+1;

setlength(NArray,lengthAr,lengthAr);

**if** first **then**

**begin**

border:=1;

**for** i:=0 **to** lengthAr-1 **do**

**for** j:=0 **to** lengthAr-1 **do**

**begin**

**if** border <= length(enterstring) **then**

NArray[i,j]:=enterstring[border]

**else**

NArray[i,j]:=' ';

border:=border+1;

**end**;

**end**;

**end**;

**procedure** createstr(CArray:TArD;lengthStrInAr:integer; **var** totalstr:string);

**var**

i,j:integer;

**begin**

totalstr:='';

**for** i:=0 **to** lengthStrInAr-1 **do**

**for** j:=0 **to** lengthStrInAr-1 **do**

totalstr:=totalstr + CArray[i,j]

**end**;

{Шифрование}

**procedure** cypher(CArray:TArD;lengthAr,sim:integer; **var** firststatus:TStat;

**var** totalstr:string; **var** direction:integer);

**Var**

status:TStat;

centr,x,y,number,step,range,rangestep:integer;

**begin**

randomize;

firststatus:=random(4)+1;

direction:=random(2); //0 против 1 по

status:=firststatus;

totalstr:='';

centr:=(sqr(lengthAR)+1) **div** 2;

x:=ceil(centr/lengthAr);

y:= centr **mod** lengthAr;

randomize;

number:=1;

{step - сколько шагов сделано в закономерности <2}

step:=0;

{range - зависимое значение от step, барьер для rangestep}

range:=1;

rangestep:=1;

totalstr:=totalstr + CArray[y-1,x-1];

**while** number<sqr(lengthAR) **do**

**begin**

**while** (rangestep<=range) **and** (number<sqr(lengthAR)) **do**

**begin**

**if** status = 1 **then** //up

dec(y)

**else**

**if** status =2 **then** //left or rigth

**begin**

**if** direction=0 **then**

dec(x)

**else**

inc(x);

**end**

**else**

**if** status = 3 **then** //down

inc(y)

**else**

**if** status = 4 **then** //rigth or left

**begin**

**if** direction=0 **then**

inc(x)

**else**

dec(x);

**end**;

totalstr:=totalstr + CArray[y-1,x-1];

inc(rangestep);

inc(number);

**end**;

inc(step);

rangestep:=1;

inc(status);

**if** step>=2 **then**

**begin**

inc(range);

step:=0;

**end**;

**if** status>4 **then**

status:=low(status);

**end**;

**end**;

**procedure** uncypher(CArray:TArD;lengthAr,sim:integer;

**var** totalstr:string;status:TStat;direction:integer);

**Var**

centr,x,y,number,step,range,rangestep:integer;

**begin**

centr:=(sqr(lengthAR)+1) **div** 2;

x:=ceil(centr/lengthAr);

y:= centr **mod** lengthAr;

number:=1;

{step - сколько шагов сделано в закономерности <2}

step:=0;

{range - зависимое значение от step, барьер для rangestep}

range:=1;

rangestep:=1;

CArray[y-1,x-1]:=totalstr[number];

inc(number);

**while** number<=sqr(lengthAR) **do**

**begin**

**while** (rangestep<=range) **and** (number<=sqr(lengthAR)) **do**

**begin**

**if** status = 1 **then** //up

dec(y)

**else**

**if** status =2 **then** //left or rigth

**begin**

**if** direction=0 **then**

dec(x)

**else**

inc(x);

**end**

**else**

**if** status = 3 **then** //down

inc(y)

**else**

**if** status = 4 **then** //rigth or left

**begin**

**if** direction=0 **then**

inc(x)

**else**

dec(x);

**end**;

CArray[y-1,x-1]:=totalstr[number];

inc(rangestep);

inc(number);

**end**;

inc(step);

rangestep:=1;

inc(status);

**if** step>=2 **then**

**begin**

inc(range);

step:=0;

**end**;

**if** status>4 **then**

status:=low(status);

**end**;

**end**;

**Begin**

writeln('Enter string');

readln(frontstring);

createarray(frontstring,true,NArray,lengthAr);

writeln('The encrypted string:');

cypher(NArray,lengthAr,length(frontstring),

firststatus,totalstr,direction);

writeln(totalstr);

createarray(totalstr,false,totalArray,lengthAr);

writeln('The decrypted string');

uncypher(totalArray,lengthAr,length(totalstr),

totalstr,firststatus,direction);

createstr(totalArray,lengthAr,totalstr);

writeln(totalstr);

readln;

**End**.