# Министерство образования Республики Беларусь

Учреждение образования

БелорусскиЙ государственный университет

информатики и радиоэлектроники

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

|  |
| --- |
|  |
|  |

# оТЧЕТ

по лабораторной работе

на тему:

Простые циклы

|  |  |  |
| --- | --- | --- |
| Выполнил  Студент гр. 451001 |  | В. А. Пузик |
| Проверил |  | Асс. Е.Е. Фадеева |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Минск, 2018

1. Теоретические сведения по теме лабораторной работы

Типы циклов:

1. По типу вложенности
   1. Внешние циклы — циклы, не имеющие родительских, не вложенные в другой цикл
   2. Вложенный цикл — циклы, вложенные в другой цикл
   3. сложный цикл — конструкция, состоящая из внешнего и как минимум одного ложенного цикла
2. По типу счетчика
   1. цикл со счетчиком — цикл, выполняющийся фиксированное кол-во раз
   2. итерационный цикл — цикл, выполняющийся до тех пор, пока выполняется определенное условие, и количество итераций может быть неопределенным заранее.

Алгоритм - система правил, четко описывающая последовательность действий, которые необходимо выполнить для решения задачи.

Свойства правильного алгоритма:

1. Дискретность - любой алгоритм состоит из отдельных операций (этапов, действий), которые выполняются дискретно (по шагам).
2. Определенность — свойство алгоритма, которое гласит, что каждый его шаг должен быть строго определен и различные толкования должны быть исключены.
3. Результативность — свойство, которое гласит, что любой алгоритм должен завершаться с результатом.
4. Массовость — свойство, которое гласит, что любой алгоритм можно успешно применять к различным наборам исходных данных в определенном формате.
5. Задание на лабораторную работу
   1. Постановка задачи  
      36. Найти последовательность из 50 нулей и единиц, в которой никакой отрезок не повторяется три раза подряд. Например, в искомой последовательности нигде не должны встречаться такие отрезки, как 000, или 101010, или 101101101.Эскиз ожидаемого результата.
   2. Эскиз ожидаемого результата  
      1100101…110
6. Выполнение
   1. Разработка алгоритма

Таблица 3.1 используемые идентификаторы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя идентификатора | назначение | Тип идентификатора | Начальное значение | Закон изменения | Имя цикла, в котором происходит изменение переменной |
| Iterating | Условие выполнения цикла MainCycle | Boolean | True | if IsCheckedPassed  Iterating := False; | MainCycle |
| IsCheckPassed | Сохраняет результат проверки для текущей попытки решения | Boolean | - | IsCheckedPassed := Check(A) | MainCycle |
| Cnt | Кол-во попыток сгенерировать решение | Integer | 0 | Inc(Cnt); | MainCycle |
| A | Текущий результат генерации | String | - | A := Solve(50); | MainCycle |
| Res | Результат полученный в функции генерации решения | String | Prev (“10”) | if Prev = '00' then  Res := Res + '1'  else if Prev = '11' then  Res := Res + '0'  else  begin  if Random < 0.5 then  Res := Res + '1'  else  Res := Res + '0';  end; | GenCycle |
| Prev | Предыдущая пара символов в функции генерации | String | “10” | Prev := Res[i] + Res[i + 1]; | GenCycle |
| L | Длина проверяемой подстроки | Integer | 1 | for L := 1 to Length(X) do | LengthCycle |
| i | Индекс начала проверяемых подстрок | Integer | 1 | for i := 1 to Length(X) - (L \* 3) do | PositionCycle |
| A1 | Первая подстрока | String | - | A1 := Copy(X, i, L); | PositionCycle |
| A2 | Втоаря подстрока | String | - | A2 := Copy(X, i + L, L); | PositionCycle |
| A3 | Третья подстрока | String | - | A3 := Copy(X, i + 2 \* L, L); | PositionCycle |
| Ans | Результат проверки | Boolean | True | if (A1 = A2) and (A2 = A3) then  Ans := false; | PositionCycle |





Для цикла MainCycle был выбран оператор while, так как заранее неизвестно сколько раз будет необходимо сгенерировать решение перед тем, как оно пройдет проверку. Цикл LengthCycle и PositionCycle и GenCycle реализованы при помощи цикла for так как сразу известно сколько итераций займет проход по каждому из вышеописанных циклов.

* 1. Текст программы и его описание

Алгоритм в общем виде должен работать следующим образом:

1. Сгенерировать попытку решения и записать ее в переменную
2. Проверить что решение валидное
3. В случае провала проверки – перейти к шагу 1
4. Вывести результат
5. конец работы алгоритма

Алгоритм проверки:

1. Запишем в результат true
2. Возьмем размер подстроки на 1 больше предыдущего (в первый раз - 1)
3. Запишем индекс первой подстроки на 0
4. Если обрабатывается размер подстроки таков, что мы не можем взять новые 3 идущие подряд подстроки для сравнения, перейдем к шагу 11
5. Увеличим текущий обрабатываем индекс первой подстроки на 1
6. Если значение (обрабатываемый индекс + обрабатываемая длина \* 3) больше, чем длина исходной строки перейдем к шагу 2
7. Запишем в переменные нужные для проверки подстроки
   1. От обрабатываемого индекса до обрабатываемого индекса + обрабатываемая длина
   2. От обрабатываемого индекса + обрабатываемая длина до обрабатываемого индекса + обрабатываемая длина \* 2
   3. От обрабатываемого индекса + обрабатываемая длина \* 2 до обрабатываемого индекса + обрабатываемая длина \* 3
8. Если ложно выражение о том, что первая строка равна второй и вторая – третьей, перейдем к шагу 10
9. Запишем в результат false
10. Перейти к шагу 4
11. Вернем результат

Алгоритм генерации решений:

1. Запишем в результат строку “10”
2. Если длина результата больше или равна значению полученному из параметров решения перейдем к шагу N
3. Если последние 2 символа ответа – «00» добавим к ответу «1»
4. Если последние 2 символа ответа – «11» добавим к ответу «0»
5. Если последние 2 символа ответа не равны «00» или «11» и случайное значение от 0 до 1 меньше 0.5, добавим к ответу «1» и перейдем к шагу 7
6. Добавим к ответу «0»
7. Запишем последние 2 символа ответа в переменную
8. Перейдем к шагу 2
9. Вернуть результат генерации

uses

SysUtils;

function Solve(Length: Integer): string;

var

Prev, Res: string;

i: Integer;

begin

Prev := '10';

Res := Prev;

for i := 2 to Length - 1 do

begin

if Prev = '00' then

Res := Res + '1'

else if Prev = '11' then

Res := Res + '0'

else

begin

if Random < 0.5 then

Res := Res + '1'

else

Res := Res + '0';

end;

Prev := Res[i] + Res[i + 1];

end;

Result := Res;

end;

function Check(const X: string): Boolean;

var

L, i: Integer;

A1, A2, A3: string;

Ans: Boolean;

begin

Ans := true;

for L := 1 to Length(X) do

begin

for i := 1 to Length(X) - (L \* 3) do

begin

A1 := Copy(X, i, L);

A2 := Copy(X, i + L, L);

A3 := Copy(X, i + 2 \* L, L);

if (A1 = A2) and (A2 = A3) then

Ans := false;

end;

end;

Result := Ans;

end;

procedure Main;

var

Iterating, IsCheckPassed: Boolean;

Cnt: Integer;

A: string;

begin

Randomize;

Iterating := True;

Cnt := 0;

while Iterating do

begin

A := Solve(50);

IsCheckedPassed := Check(A)

Inc(Cnt);

if IsCheckedPassed then

begin

Writeln(Format('Success in %s, Result is: %s', [IntToStr(Cnt), A]));

Iterating := False;

end;

end;

Readln;

end;

begin

Main;

end.

* 1. Тестирование и отладка программы

Таблица 3.2 Прохождение тестов программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Специфика тестирования | Номер теста | Вводимые данные | Ожидаемый результат | Полученный результат |
| Корректность работы программы на небольших значениях (11) | 1 | -  (запуск функции генерации для 10 символов) | Нет повторов | 1001100110 |

* 1. Итоговый текст программы

uses

SysUtils;

function Solve(Length: Integer): string;

var

Prev, Res: string;

i: Integer;

begin

Prev := '10';

Res := Prev;

for i := 2 to Length - 1 do

begin

if Prev = '00' then

Res := Res + '1'

else if Prev = '11' then

Res := Res + '0'

else

begin

if Random < 0.5 then

Res := Res + '1'

else

Res := Res + '0';

end;

Prev := Res[i] + Res[i + 1];

end;

Result := Res;

end;

function Check(const X: string): Boolean;

var

L, i: Integer;

A1, A2, A3: string;

Ans: Boolean;

begin

Ans := true;

for L := 1 to Length(X) do

begin

for i := 1 to Length(X) - (L \* 3) do

begin

A1 := Copy(X, i, L);

A2 := Copy(X, i + L, L);

A3 := Copy(X, i + 2 \* L, L);

if (A1 = A2) and (A2 = A3) then

Ans := false;

end;

end;

Result := Ans;

end;

procedure Main;

var

Iterating, IsCheckPassed: Boolean;

Cnt: Integer;

A: string;

begin

Randomize;

Iterating := True;

Cnt := 0;

while Iterating do

begin

A := Solve(50);

IsCheckedPassed := Check(A)

Inc(Cnt);

if IsCheckedPassed then

begin

Writeln(Format('Success in %s, Result is: %s', [IntToStr(Cnt), A]));

Iterating := False;

end;

end;

Readln;

end;

begin

Main;

end.