



Урок 5. Оператор цикла с параметром

Задачи урока: Содействовать формированию

Создавать условия для воспитания

Способствовать развитию

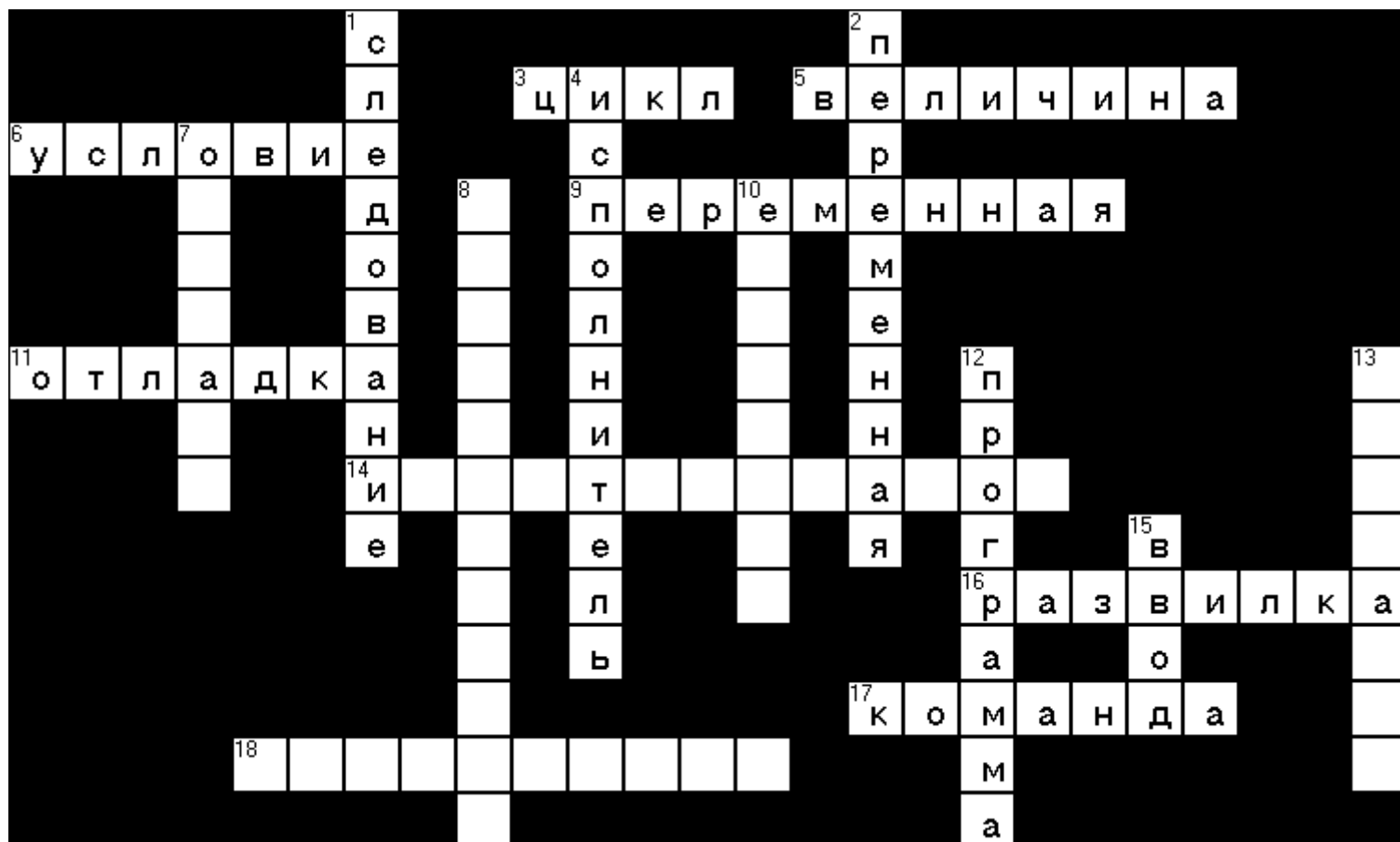
Учащиеся должны знать: алгоритмическую конструкцию «повторение»

Учащиеся должны уметь: реализовывать на языке программирования алгоритмы с повторением

ТЕХНОЛОГИЧЕСКАЯ КАРТА УРОКА

Этап урока	Вре- мя	Теория	Практика
1. Организацион-ны й момент.	1	Вход в урок. Создание условий для положительной мотивации учащихся. Входная рефлексия.	
2. Проверка домашнего задания	2-8	Загрузка сервера тестирования: MyTestServer	
		Раздать файл:  Т 4.mtf (вопросов-, время- мин.)	
		 § (?), стр.	
		Загрузка сетевой программы: NetOp School	
3. Изучение нового материала	9-15	Вопросы:	
4. Закрепление знаний	16-2 5	 § (?), стр.	 ПР 5 №1 ⁰ ,2 ⁰
5. Контроль знаний	26-4 2	 ПР 5 №3-5	
		Загрузка сервера тестирования: MyTestServer	
		Раздать файл:  Т5 .mtf (вопросов-, время- мин.)	

		 К 5
6. Итог урока	43-4 5	Подведение итогов урока. Выставление оценок в журнал и дневники. Ответы на вопросы.
7. Рефлексия		
8. Домашнее задание		§ 3, (3.1-3.2), упр. 3а. Завершение урока.



- Развивать умение видеть циклическую структуру и оценивать результат выполненных действий.
- Воспитание познавательной потребности.

План урока:

- I. Организационно-мотивирующий этап.
- II. Повторение + проверка знаний по пройденному материалу.
- III. Объяснение нового материала
- IV. Закрепление пройденного материала (физкультминутка).
- V. Контрольно-оценочный этап.
- VI. Рефлексия.
- VII. Д/з.

ХОД УРОКА

I. Организационно-мотивирующий этап.

Повторение + проверка знаний по пройденному материалу.

1. Учащиеся проходят тестирование на компьютерах. Ответить на вопросы теста, указав номер правильного ответа/

Вариант 1.

Вопрос 1

Какие ключевые слова не используются для записи команды ветвления? (3)

1. else
2. if
3. and
4. then

Вопрос 2.

Как выглядит полная форма команды ветвления? (2)

1. if ... else ...
2. if ... then ... else ...
3. if ... then ...
4. нет правильного ответа

Вопрос 3.

Как выглядит сокращённая форма команды ветвления? (3)

1. if ... else ...
2. if ... then ... else ...
3. if ... then ...
4. нет правильного ответа

Вопрос 4.

Какая из команд записана неправильно? (2)

1. if a>0 then a:=0;
2. if a:=0 then a>0;
3. if 2*2<>5 then writeln(2*2);
4. нет правильного ответа

Вопрос 5.

Какая из команд записана правильно? (2)

1. if (x<0) and (y<0) then y*x>0;
2. if x=0 and a>0 then b:=0 else b:=x;
3. if x=0 and a>0 then b:=0; else b:=x;
4. нет правильного ответа

Вопрос 6.

Что будет напечатано на экране в результате выполнения данного фрагмента программы (4)
a:=6;

if a mod 2 = 0 then write(a+2) else write(a-2);

1. 4
2. 10
3. 12
4. 8

Вопрос 7.

Что будет напечатано на экране в результате выполнения данного фрагмента программы (3)
a:=12;

if a div 2 >= 6 then b:=a mod 5 else b:=a div 3;

write(a-b);

1. 0
2. 8
3. 2
4. 10

Вопрос 8.

Вставьте пропущенную логическую операцию так, чтобы значение переменной z после выполнения заданного фрагмента программы стало равным 9. (2)

a:=5; b:=8; c:=11;

if (a>b) ... (b<c) then z:=a+4 else z:=b+3;

1. and
2. or
3. not
4. не знаю

Вопрос 9.

Вставьте пропущенную логическую операцию так, чтобы значение переменной z после выполнения заданного фрагмента программы стало равным 3. (3)

a:=15; b:=7; c:=4;

if (a>b) ... (b<c) then z:=a+c else z:=b-c;

1. or
2. not
3. and
4. не знаю

Вопрос 10.

Вставьте пропущенную логическую операцию так, чтобы значение переменной z после выполнения заданного фрагмента программы стало равным 5. (2)

a:=1; b:=8; c:=6;

if (a<=c) ... (b<=c) then z:=c-a else z:=c-b;

1. not
2. or
3. and
4. не знаю

Вариант 2.

Вопрос 1.

Когда используется алгоритм с ветвлением? (2)

1. Когда у нас все время повторяется какое то действие.
2. Когда нам в зависимости от ситуации необходимо поступить так или иначе.
3. Когда пользуемся линейными алгоритмами

Вопрос 2.

Установите соответствие:(1-а, 2-в, 3-с).

1. Если
2. то
3. иначе

- a. if
- в. then
- c. else

Вопрос 3.

Что будет написано в окне вывода при выполнении программы если a=5 а b=8? (8)

**if a>b then
write('a')
else write('b');**

1. 5
2. a>b
3. a<b
4. b=8
5. 8

Вопрос 4.

Укажите неправильно заданный фрагмент условия.(3)

- 1.if a>b then ...
- 2.if a=b then...
- 3.if a:=b then...
- 4.if a<b then...

Вопрос 5

Какой результат вы увидите на экране при выполнении программы: (4)

```
var n,m,rez integer;  
begin  
n:=1;  
m:=2;  
if n>m  
    then rez:=n/m  
    else rez:=m/n;  
write(rez);  
end.
```

1. 1/2
2. 2/1
3. 2
4. Программа не выполнится из-за ошибки

Вопрос 6

Даны два числа x и y, вам нужно найти большее из них. Ваши действия: (4)

1. if x>y write(x);
2. if x<y then write(y);
3. if x>y then write(x);_CRLF_else write(y);
4. if x>y then write(x)_CRLF_else write(y);

Вопрос 7

Определите тип условия: (2)

```
if a>b  
then write('a>b')  
else if a=b  
    then write('a=b')  
    else write('a<b');
```

- 1.Простое условие
- 2.Составное условие
- 3.Сложное условие
- 4.Трудное условие

Вопрос 8

Выберите команды, которые относятся к алгоритмической конструкции "ветвление". (2,4)

- 1.var
- 2.if
- 3.begin
- 4.else
- 5.write

Вопрос 9

Как сделать так, чтобы программа выполнялась в отдельном окне? (1)

- 1.Подключить библиотеку CRT;
- 2.Подключить библиотеку GraphABC;

3. Нажать выполнить в отдельном окне

Вопрос 10

Если условие ложно, то ... (2)

1. Выполняется то, что идет после команды then

2. Выполняется то, что идет после команды else

2. Один ученик работает у доски (записывает решение домашней задачи)

После выполнения теста проверить фронтально решение задачи.

3. Фронтальный опрос.

- На прошлых уроках мы занимались ветвлением . Сколько раз выполнится действие (действия) и от чего это зависит?
- Что такое цикл и где в нашей жизни мы с ним встречаемся? (многократное повторение определенных действий, например, смена дня и ночи, времена года...) . Цикл, дословно с греческого – круг.

Сегодня на уроке мы познакомимся с организацией цикла в Паскале.

III. Объяснение нового материала

1. **Цикл** является одной из базовых алгоритмических конструкций и *представляет собой последовательность действий, которая выполняется неоднократно, до тех пор пока выполняется некоторое условие*. Саму последовательность повторяющихся действий называют **телом цикла**.

Циклы позволяют записать действия в компактной форме. Например, для *вычисления суммы первых десяти натуральных чисел* можно выбрать простое решение и записать вычисление в строчку, употребив 9 операций сложения:

$$\text{Sum}:=1+2+3+4+5+6+7+8+9+10;$$

А если число слагаемых достаточно велико (1000 или 10000)? Представьте себе программу с оператором присваивания, который занимает несколько страниц и содержит 999 и более операций сложения! Очевидно, такое решение в этом случае не подходит.

Рассмотрим подробнее *алгоритм вычисления суммы первых 5 натуральных чисел по шагам*. В переменную Sum будем накапливать сумму чисел, в переменную i будем записывать очередное натуральное число.

0 шаг: $\text{sum}_0 = 0$

1 шаг: $i=1$ $\text{sum}_1 = \text{sum}_0 + i = 0 + 1 = 1$

2 шаг: $i=2$ ($i=i+1=1+1$) $\text{sum}_2 = \text{sum}_1 + i = 1 + 2 = 3$

3 шаг: $i=3$ ($i=i+1=2+1$) $\text{sum}_3 = \text{sum}_2 + i = 3 + 3 = 6$

4 шаг: $i=4$ ($i=i+1=3+1$) $\text{sum}_4 = \text{sum}_3 + i = 6 + 4 = 10$

5 шаг: $i=5$ ($i=i+1=4+1$) $\text{sum}_5 = \text{sum}_4 + i = 10 + 5 = 15$

Нетрудно заметить, что при вычислении суммы повторяются всего две операции, причем в определенном порядке:

1. Прибавить слагаемое к ранее полученной сумме:
2. Увеличить на 1 значение слагаемого.

Следовательно, задачу можно решить с помощью следующего алгоритма:

1. Присвоить переменной Sum значение, равное 0 ($\text{Sum}:=0$).
2. Присвоить переменной i (слагаемое) значение, равное 1 ($i:=1$).
3. Добавить к сумме значение слагаемого i ($\text{Sum}:= \text{sum}+i$).

4. Увеличить i на 1 ($i:=i+1$).

5. Повторить шаги 3 и 4.

Повторив операции 3 и 4 5 либо 9 либо 999 (или более) раз, получим требуемую сумму. Это пример алгоритмической конструкции «Цикл (повторение)».

2. На языке Паскаль повторение некоторой последовательности действий известное число раз выполняет оператор цикла **for**. В общем виде оператор **for** представлен в виде:

For <переменная цикла>:= <начальное значение> **to** <конечное значение> **do** <оператор>;

Кратко: **For** $i:=N1$ **to** $N2$ **do** <оператор>;

В представленных формах оператора цикла **For.....do** – заголовок цикла, <оператор> – тело цикла. Тело цикла может быть простым или составным оператором. *Переменная цикла, ее начальное и конечное значения должны принадлежать к одному и тому же типу данных (чаще всего это целочисленный int integer, но могут быть и другие типы, кроме вещественного).*

Блок-схема цикла с заданным числом повторений **For...to** может иметь следующий вид :



Пример 1. Сумма первых десяти натуральных чисел

```
program summa;
uses CRT;
var i,sum:integer;
begin
  write('n=');
  readln(n);
  sum:=0;
  for i:=1 to n do
    begin
      writeln('i=',i);
      writeln('Sum =',sum,'+',i);
      sum:=sum+i;
    end;
  writeln('Sum =',sum);
end.
```

Результат выполнения программы:

Sum =55

В программе **summa** (пример 1) переменная i является целочисленной управляющей переменной цикла **for** (или его счетчиком). В заголовке оператора **for** указаны начальное и конечное значения переменной цикла i . В данном случае начальное значение переменной цикла равно 1. При первом выполнении тела цикла $i:=1$, при втором - $i:=2$, и т.д. При последнем выполнении тела цикла $i:=10$.

Каждый раз перед выполнением тела цикла текущее значение переменной цикла *i* сравнивается с конечным значением. После каждого выполнения тела цикла переменная *i* увеличивается на 1. Как только *i* превысит конечное значение, выполнение цикла прекращается и осуществляется переход на следующий после `for` оператор программы.

4. Выполнить программу за компьютером. Обратить внимание на изменение счётчика в команде цикла.

5. Изменить в программе `summa` (пример 1) *operator for i:=1 to n do*

на operator for i:=n to 1 downto. В этом случае программа примера 1 будет выглядеть следующим образом:

Пример 2. Сумма первых N натуральных чисел

```
program summa_N;  
  uses CRT;  
  var i,sum,n:integer;  
  begin  
    write ('n=');  
    readln(n);  
    sum:=0;  
    for i:=10 downto 1 do  
      begin  
        writeln('i=',i);  
        writeln('Sum =',sum,'+',i);  
        sum:=sum+i;  
      end;  
    writeln('Sum =',sum);  
  end.
```

Результат выполнения программы при n=10:

```
n=10  
Sum =55
```

6. Проанализировать программу и определить сходство и отличие в формах записи команды цикла.

В общем виде оператор `for` может быть представлен в двух формах. Первую форму уже рассмотрели ранее, а теперь рассмотреть вторую форму записи.

Первая форма последовательно наращивает переменную цикла, увеличивая ее каждый раз на 1:

Вторая форма последовательно уменьшает переменную цикла на 1:

For <переменная цикла>:= <начальное значение> **downto** <конечное значение> **do** <оператор>;

Кратко: `For i:=N1 downto N2 do <оператор>;` (N1> N2)

Важно помнить:

1. Не рекомендуется изменять управляющую переменную цикла в теле цикла с помощью команды

присваивания. Это может привести к «зацикливанию» программы (бесконечному повторению тела цикла) либо к неверным результатам выполнения программы.

n:=10;	n:=30;
for i:=1 to n do	for i:=n downto 5 do
i:=i+1;	writeln(i);

2. Если в заголовке оператора `for..to` начальное значение переменной цикла больше конечного значения, то тело цикла не выполнится ни разу.

```
n:=10;
for i:=12 to n do
s:=s+i;
```

12>10 — тело цикла не выполнится ни разу.

3. Если в заголовке оператора `for..downto` начальное значение переменной цикла меньше конечного значения, то тело цикла не выполнится ни разу.

```
n:=10;
for i:=2 downto n do
s:=s+1;
```

2<10 — тело цикла не выполнится ни разу.

4. При выходе из цикла текущее значение переменной цикла больше равно конечному значению в операторе `for..to` и наоборот в операторе `for..downto`.

n:=10;	n:=10;
for i:=1 to n do	for i:=n downto 3 do
s:=s+i;	s:=s+i;

Конечное значение i=11.

Конечное значение i=2.

5. Телом цикла может быть не один оператор, а группа операторов (составной оператор), которую обязательно надо заключать в операторные скобки `begin... end`.

```
n:=10;
for i:=1 to n do
begin
sum:=sum+i;
writeln('Sum =',sum);
end;
```

Начальным значением управляющей переменной цикла `for..to` не обязательно должна быть единица. Чтобы узнать, сколько раз выполнится тело цикла `for`, нужно найти разность между крайними значениями переменной цикла (по модулю) и прибавить к результату 1.

IV. Закрепление пройденного материала.

1. Вывести на экран площади 5 квадратов, если сторона каждого последующего квадрата на 10 больше предыдущего.

uses CRT;

```

var i,a,s:integer;
begin
write('сторона меньшего квадрата a=');
readln (a);
for i:=1 to 5 do
begin
a:=a+5;
s:=a*a;
write(s);
end;
end.

```

2. Введите количество учащихся в группе по информатике и четвертную отметку каждого учащегося. Вычислите средний балл успеваемости по информатике в группе за четверть.

```

uses crt;
var i,k,b,sum,sball_c : integer;
    sball : real;
begin
writeln('Введите количество учащихся в группе: ');
readln(k);
sum := 0;
writeln('Введите отметки по информатике за четверть');
for i := 1 to k do
begin
write(i,': ');
readln(b);
sum := sum+b;
end;
sball := sum/k;
writeln('Средний балл группы за четверть: ',sball:5:2);
end.

```

V. Контрольно – оценочный этап

Ответить ДА + или НЕТ -

1. В теле цикла со счетчиком имеется возможность использовать несколько, а не один оператор. (+)
2. Для использования цикла со счетчиком необязательно знать и невозможно вычислить количество повторений. (-)
3. Если в заголовке оператора for..to начальное значение переменной цикла больше конечного значения, то тело цикла не выполнится ни разу.(+).
4. Для того, чтобы тело цикла выполнялось при уменьшающихся значениях параметрах цикла, необходимо использовать в записи данного оператора слово DOWNTO. (+)

VI. Рефлексия

- Что нового вы узнали на сегодняшнем уроке?
- Решение какой задачи было для вас наиболее сложным?
- С какими именно трудностями вы столкнулись при решении этой задачи?
- Какие задания показались вам простыми?

VII. ДЗ.

1) § 3,п.1,2. Составить конспект.

2) Решить задачи

1. Вывести кубы натуральных чисел от 7 до 12 в порядке возрастания.

2. Вывести кубы натуральных чисел от 7 до 12 в порядке убывания.

3. На уборке картофеля работали N человек. Введите массу картофеля, собранного каждым работающим, и вычислите массу собранного картофеля.

```

uses crt;
var i,k : integer;
    b,sum : real;
begin
writeln('Введите количество работающих на уборке: ');

```

```
readln(k);  
sum := 0;  
writeln('Введите массу убранный картофеля');  
for i := 1 to k do  
begin  
  write(i, ' ');  
  readln(b);  
  sum := sum+b; {находим массу убранный картофеля}  
end;  
writeln('Всего убрали ',sum,' кг картофеля');  
end.
```


Тема: Алгоритмы с повторением.Оператор цикла с параметром		Тема: Алгоритмы с повторением.Оператор цикла с параметром	
Общий вид <pre> 1) For i:=k1 to k2 do <оператор>; 3) For i:=k1 to k2 do begin <оператор 1>; <оператор 2>; ... <оператор n>; end;</pre> Где i – параметр цикла k1 - начальное значение цикла k2 - конечное значение цикла		Общий вид <pre> 2) For i:=k1 downto k2 do <оператор>; 4) For i:=k1 downto k2 do begin <оператор 1>; <оператор 2>; ... <оператор n>; end;</pre> Где i – параметр цикла k1 - начальное значение цикла k2 - конечное значение цикла	
Работа цикла FOR Начальное значение k1 сравнивается с конечным k2. Если условие k1<=k2 истинно, то выполняется тело цикла. Затем значение k1 увеличивается на 1 и вновь сравнивается с k2, если условие k1<=k2 истинно, то снова выполняется тело цикла. Процесс продолжается до тех пор пока k1 не превысит k2. В случаях 2 и 3 происходит уменьшения параметра цикла с каждым шагом на 1. Тело цикла выполняется при условии когда k1>=k2.		Работа цикла FOR Начальное значение k1 сравнивается с конечным k2. Если условие k1<=k2 истинно, то выполняется тело цикла. Затем значение k1 увеличивается на 1 и вновь сравнивается с k2, если условие k1<=k2 истинно, то снова выполняется тело цикла. Процесс продолжается до тех пор пока k1 не превысит k2. В случаях 2 и 3 происходит уменьшения параметра цикла с каждым шагом на 1. Тело цикла выполняется при условии когда k1>=k2.	
Пример 1 (Чему будет равна величина S после выполнения фрагмента программы) S:=0; For i:=1 to 4 do S:=s+2*i; <u>1 шаг</u> <u>1<=4 (истинно)</u> S:=0+2*1=2; i:=1+1=2; <u>2 шаг</u> <u>2<=4 (истинно)</u> S:=2+2*2=6; i:=2+1=3; <u>5 шаг</u> <u>5<=4 (ЛОЖЬ)</u>		Пример 1 (Чему будет равна величина S после выполнения фрагмента программы) S:=0; For i:=1 to 4 do S:=s+2*i; <u>1 шаг</u> <u>1<=4 (истинно)</u> S:=0+2*1=2; i:=1+1=2; <u>2 шаг</u> <u>2<=4 (истинно)</u> S:=2+2*2=6; i:=2+1=3; <u>5 шаг</u> <u>5<=4 (ЛОЖЬ)</u>	
<u>3 шаг</u> <u>3<=4 (истинно)</u> S:=6+2*3=12; i:=3+1=4; <u>4 шаг</u> <u>4<=4 (истинно)</u> S:=12+2*4=20; i:=4+1=5; Ответ S=20		<u>3 шаг</u> <u>3<=4 (истинно)</u> S:=6+2*3=12; i:=3+1=4; <u>4 шаг</u> <u>4<=4 (истинно)</u> S:=12+2*4=20; i:=4+1=5; Ответ S=20	
Задача 2 Составить программу нахождения частичной суммы ряда чисел $S_n = 1 + \frac{1}{3} + \frac{1}{5} + \frac{1}{7} + \dots + \frac{1}{2n-1}$ Задача 3 Жители островов Чунга и Чанга один раз в год по праздникам обмениваются драгоценностями. Жители острова Чунга привозят половину своих драгоценностей на остров Чанга, а жители Чанга привозят треть своих драгоценностей на остров Чунга. Какая часть драгоценностей будет находиться на островах через M лет, если первоначально их у Чунга было X, а у Чанга Y		Задача 2 Составить программу нахождения частичной суммы ряда чисел $S_n = 1 + \frac{1}{3} + \frac{1}{5} + \frac{1}{7} + \dots + \frac{1}{2n-1}$ Задача 3 Жители островов Чунга и Чанга один раз в год по праздникам обмениваются драгоценностями. Жители острова Чунга привозят половину своих драгоценностей на остров Чанга, а жители Чанга привозят треть своих драгоценностей на остров Чунга. Какая часть драгоценностей будет находиться на островах через M лет, если первоначально их у Чунга было X, а у Чанга Y	

--	--