School:		
Date:	**Teacher's name:**	
Grade:	**Number present:**	**absent:**
Topic of the lesson: Conditional operator. Multiple branching		
Learning objective(s) that this lesson is contributing to	To acquaint students with the conditional operator and multiple branching by applying them when compiling programs in the Python programming language.	
Lesson objectives	**All learners will be able to**: • уметь правильно записывать условный оператор и множественное ветвление для решения конкретных задач; • **Most learners will be able to**: Apply a conditional operator and multiple branches when compiling a program for solving problems; • **Some learners will be able to:** be able to find errors in the compiled program.	
Assessment Criteria	Student: - explains the principle of the conditional operator and multiple branching; - fixes errors in the program with a conditional statement and multiple branching; - independently develops an algorithm and program for solving the problem using a conditional operator or multiple branches	
Value links	education of an emotionally positive orientation to practical activities, interest in computer science, personal responsibility for the results of their work.	
Previous learning	Data types. Variable definition Logical expressions	
Cross curricular links	Computer science, mathematics	

Time	Planned activities	Resources
Beginning __ min	Organizing time. Motivation for learning activities. Goal setting. **Knowledge Update** In the last lesson, you became familiar with data types. Definition of a variable, by logical expressions. Give your examples of tasks.	

Middle __ min	Open notebooks and write down the topic of the lesson: "Conditional if statement".
	
	In the last lesson, we learned how to write linear Python programs. Today we are going to learn the "branching" or "conditional if statement" construct.
	
	If translated into Russian, the construction of a conditional statement means the following:
	
	If the <condition is met> do: some action.
	
	For instance:
	
	`if a>b:`
	`    print(a)`
	
	"If a is greater than b, then print a."
	
	Or:	

```
if x==y:
    z=x+y
  z=z*z
```

“If x is equal to y, then z assign the value x + y, and square z.”

Indentation is important! They are part of the code. Actions will be performed only if they are all indented, and with the same number of indents. By default, the Python community makes 4 indents.

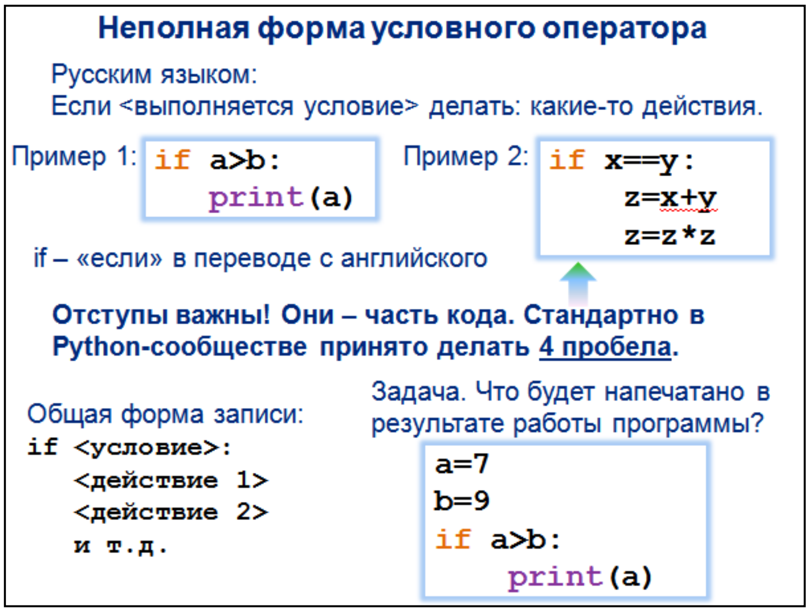
The general form of writing an incomplete form of a conditional statement:

```
if < condition >:
     < act 1>
     < act 2>
```

Task. What will be printed as a result of the program?

```
a=7
b=9
if a>b:
    print(a)
```

(Answer: nothing)



Неполная форма условного оператора

Русским языком:
Если <выполняется условие> делать: какие-то действия.

Пример 1:
```
if a>b:
    print(a)
```

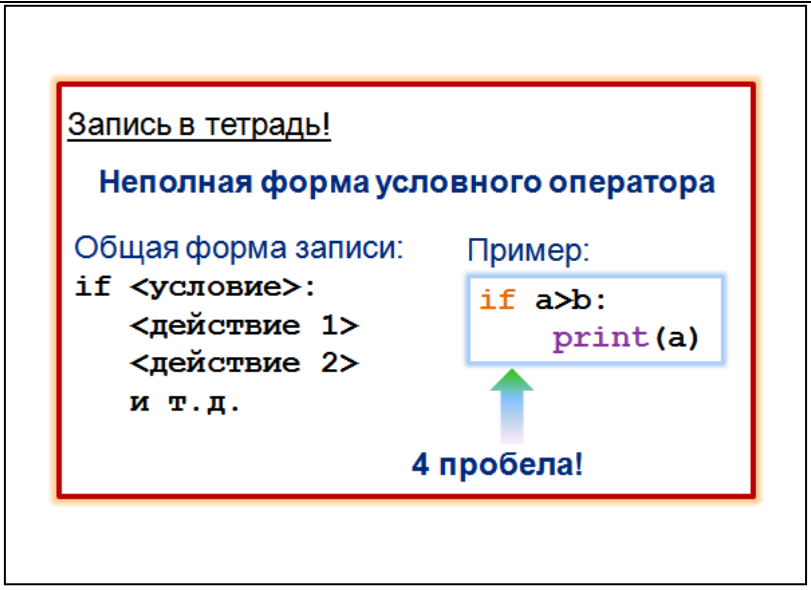
Пример 2:
```
if x==y:
    z=x+y
    z=z*z
```

if – «если» в переводе с английского

Отступы важны! Они – часть кода. Стандартно в Python-сообществе принято делать 4 пробела.

Общая форма записи:
```
if <условие>:
   <действие 1>
   <действие 2>
   и т.д.
```

Задача. Что будет напечатано в результате работы программы?
```
a=7
b=9
if a>b:
    print(a)
```



Запись в тетрадь!

Неполная форма условного оператора

Общая форма записи:

```
if <условие>:
   <действие 1>
   <действие 2>
   и т.д.
```

Пример:

```
if a>b:
    print(a)
```

4 пробела!

This was an incomplete form of a conditional statement. But the conditional operator also has a full form. In Russian, it sounds like this:

If <condition is fulfilled>: do some action. Otherwise: do other things.

Otherwise, it means "if the condition is not met."

For example:

```
if a>b:
    print(a)
else:
    print(b)
```

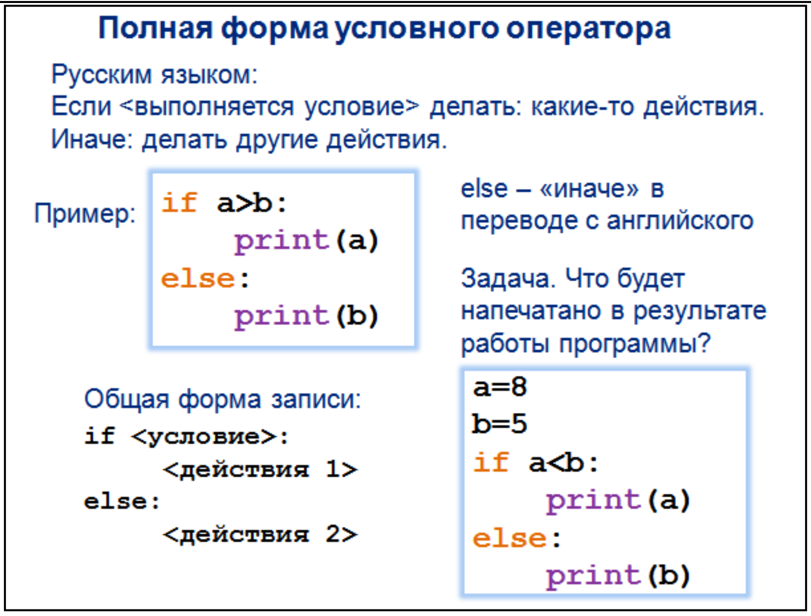
“If a is greater than b, then print a, otherwise print b.

The general form of writing an incomplete form of a conditional statement:

```
if < condition >:
< actions 1>
else:
     < actions 2>
```

Задача. Что будет напечатано в результате работы программы?

```
a=8
b=5
if a<b:
    print(a)
else:
    print(b)
```



Полная форма условного оператора

Русским языком:
Если <выполняется условие> делать: какие-то действия.
Иначе: делать другие действия.

Пример:

```
if a>b:
    print(a)
else:
    print(b)
```

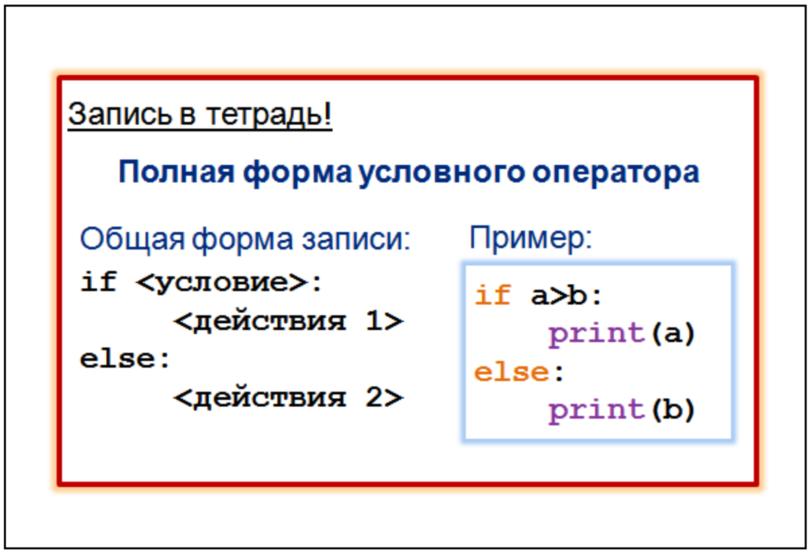
else – «иначе» в переводе с английского

Задача. Что будет напечатано в результате работы программы?

```
a=8
b=5
if a<b:
    print(a)
else:
    print(b)
```

Общая форма записи:

```
if <условие>:
      <действия 1>
else:
      <действия 2>
```



Запись в тетрадь!

Полная форма условного оператора

Общая форма записи:

```
if <условие>:
      <действия 1>
else:
      <действия 2>
```

Пример:

```
if a>b:
    print(a)
else:
    print(b)
```

Often there are tasks with a large number of conditions and actions that need to be performed when these conditions are met. The if-else construct is not enough, and then the elif operator comes to the rescue. It is explained in Russian as follows:

If <condition 1 is satisfied>: do such and such actions. Otherwise, if <condition 2 is satisfied>: do other actions.

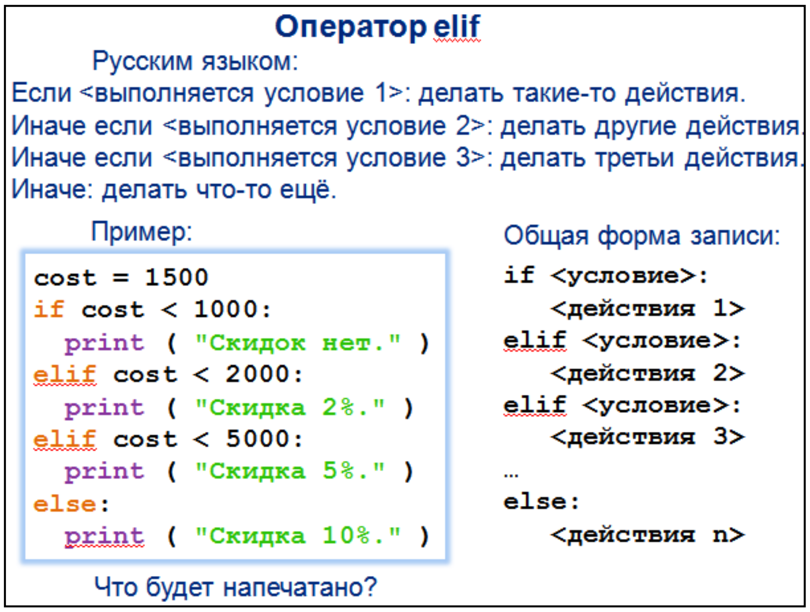
Otherwise, if <condition 3 is satisfied>: do the third action.

Otherwise: do something else.

The latter "otherwise" means "if none of the above actions are performed." The presence of "otherwise" is not necessary. For example:

```
cost = 1500
if cost < 1000:
    print ( " No discounts " )
elif cost < 2000:
    print ( " A discount 2%" )
elif cost < 5000:
    print ( " A discount 5%" )
else:
    print ( " A discount 10%" )
```

What will be printed as a result of the program? (Answer: Discount)



Оператор elif

Русским языком:

Если <выполняется условие 1>: делать такие-то действия.
Иначе если <выполняется условие 2>: делать другие действия.
Иначе если <выполняется условие 3>: делать третьи действия.
Иначе: делать что-то ещё.

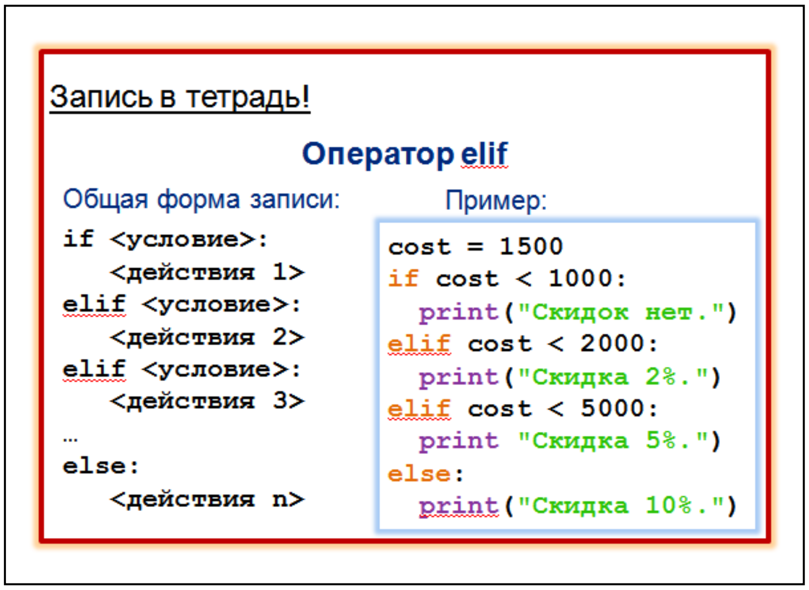
Пример:

```
cost = 1500
if cost < 1000:
  print ( "Скидок нет." )
elif cost < 2000:
  print ( "Скидка 2%." )
elif cost < 5000:
  print ( "Скидка 5%." )
else:
  print ( "Скидка 10%." )
```

Общая форма записи:

```
if <условие>:
    <действия 1>
elif <условие>:
    <действия 2>
elif <условие>:
    <действия 3>
…
else:
    <действия n>
```

Что будет напечатано?



Запись в тетрадь!

Оператор elif

Общая форма записи:

```
if <условие>:
    <действия 1>
elif <условие>:
    <действия 2>
elif <условие>:
    <действия 3>
…
else:
    <действия n>
```

Пример:

```
cost = 1500
if cost < 1000:
  print("Скидок нет.")
elif cost < 2000:
  print("Скидка 2%.")
elif cost < 5000:
  print "Скидка 5%.")
else:
  print("Скидка 10%.")
```

Signs of relationship:

> more

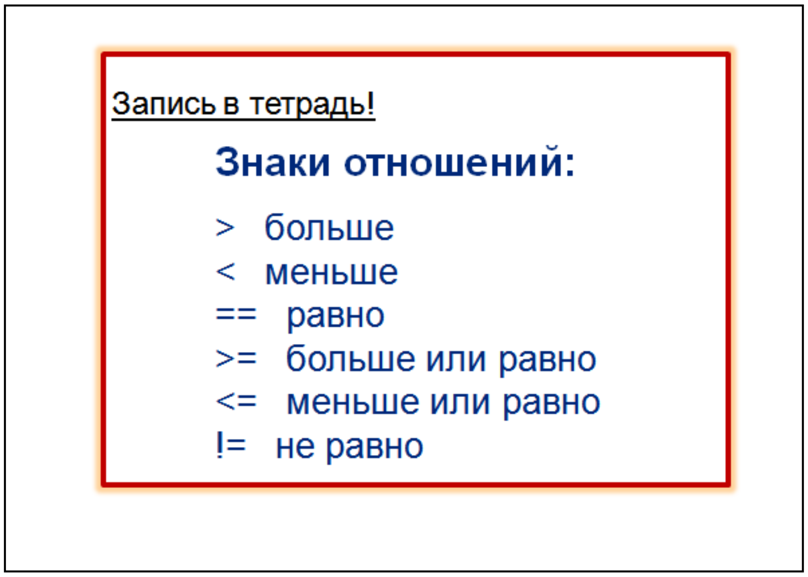
<less

== equals

> = greater than or equal

<= less than or equal

! = not equal



Запись в тетрадь!

Знаки отношений:

> больше

< меньше

== равно

>= больше или равно

<= меньше или равно

!= не равно

Difficult conditions.

To make a difficult condition, the following operators are used:

and - and

or - "or"

not - not

For example:

```
if a>0 and a<10 or a==100:
    print(a)
```

Will a be printed if a is 7? And if a is equal to 20?

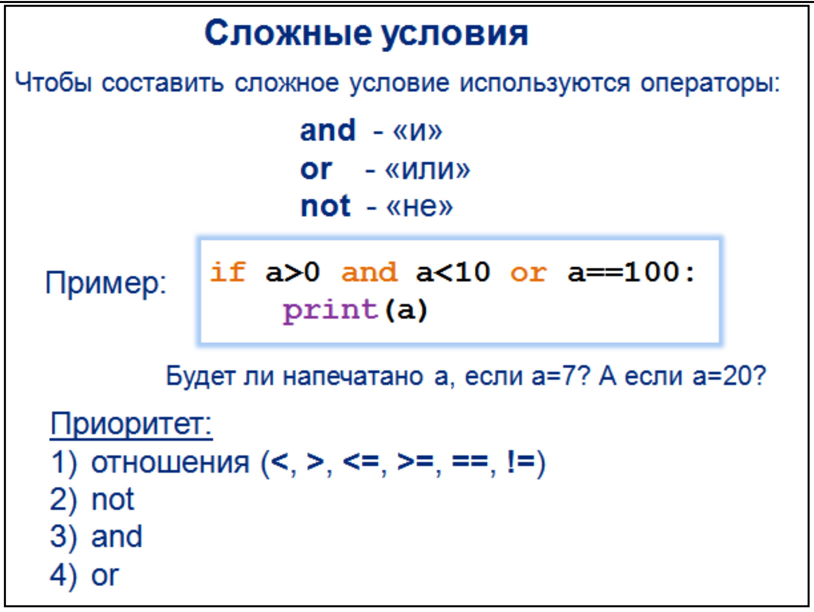
A priority :

1) relations (<,>, <=,> =, ==,! =)

2) not ("NOT")

3) and (“And”)

4) or ("OR")



Сложные условия

Чтобы составить сложное условие используются операторы:

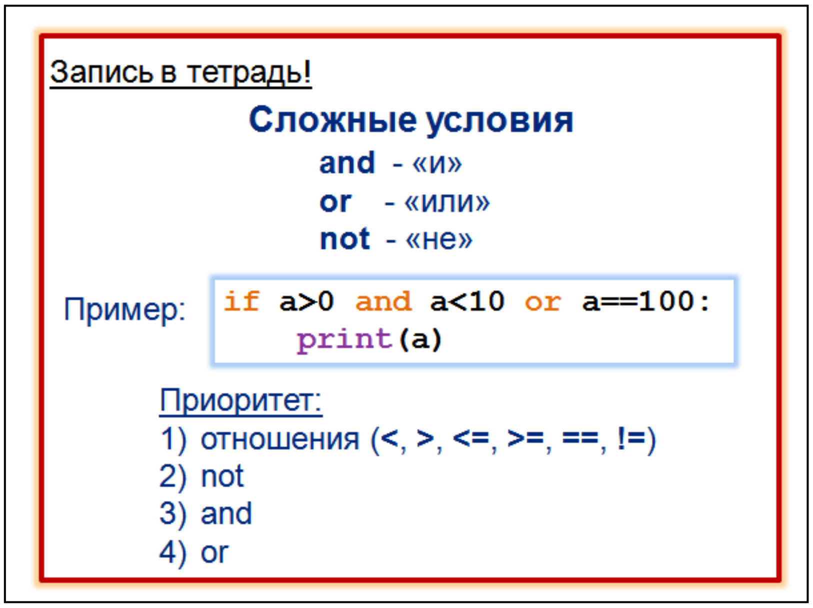
and - «и»
or - «или»
not - «не»

Пример:

```
if a>0 and a<10 or a==100:
    print(a)
```

Будет ли напечатано a, если a=7? А если a=20?

Приоритет:
1) отношения (<, >, <=, >=, ==, !=)
2) not
3) and
4) or



Запись в тетрадь!

Сложные условия

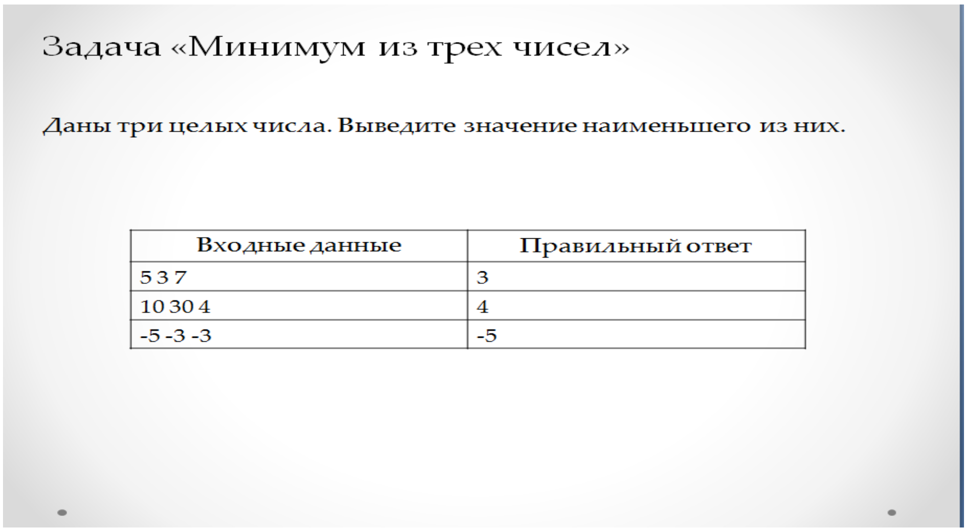
and - «и»
or - «или»
not - «не»

Пример:

```
if a>0 and a<10 or a==100:
    print(a)
```

Приоритет:
1) отношения (<, >, <=, >=, ==, !=)
2) not
3) and
4) or

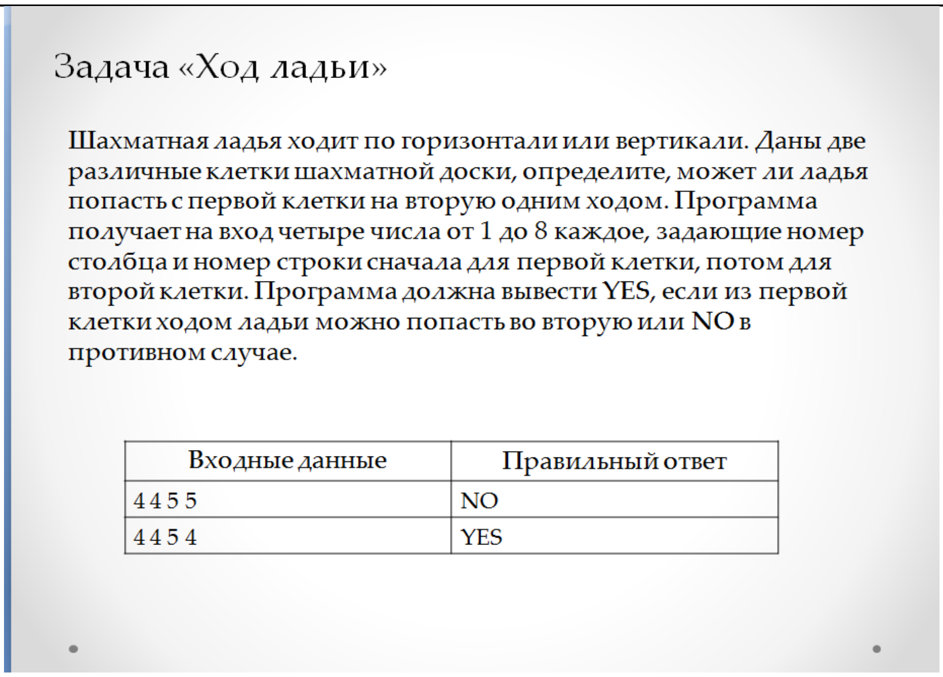
Share into 4 groups and solve problems in groups



Задача «Минимум из трех чисел»

Даны три целых числа. Выведите значение наименьшего из них.

Входные данные	Правильный ответ
5 3 7	3
10 30 4	4
-5 -3 -3	-5

	 **Задача «Ход ладьи»** Шахматная ладья ходит по горизонтали или вертикали. Даны две различные клетки шахматной доски, определите, может ли ладья попасть с первой клетки на вторую одним ходом. Программа получает на вход четыре числа от 1 до 8 каждое, задающие номер столбца и номер строки сначала для первой клетки, потом для второй клетки. Программа должна вывести YES, если из первой клетки ходом ладьи можно попасть во вторую или NO в противном случае. Входные данные | Правильный ответт 4 4 5 5 | NO 4 4 5 4 | YES	
End **__ min**	**Reflection**: 1. What did we learn today? 2. What were your difficulties in completing the tasks? 3. What did you do? 4. What did you fail? 5. How can this be fixed? **Homework** **Problem №5 page 55**	

Differentiation – how do you plan to give more support? **How do you plan to challenge the more able learners?**	**Assessment – how are you planning to check learners' learning?**	**Health and Safety**
More capable students can act as consultants on a new topic. Help weak students. Improve your projects.	Self-esteem at every stage of the lesson Evaluation of each of the tasks solved, scoring.	Compliance with the regulations when working on a computer. Active activities.