

Scalar Data
Numbers
Basic I/O
File handles

Podstawy programowania

Jolanta Bachan
2007-11-06

Implement the algorithms, cont.

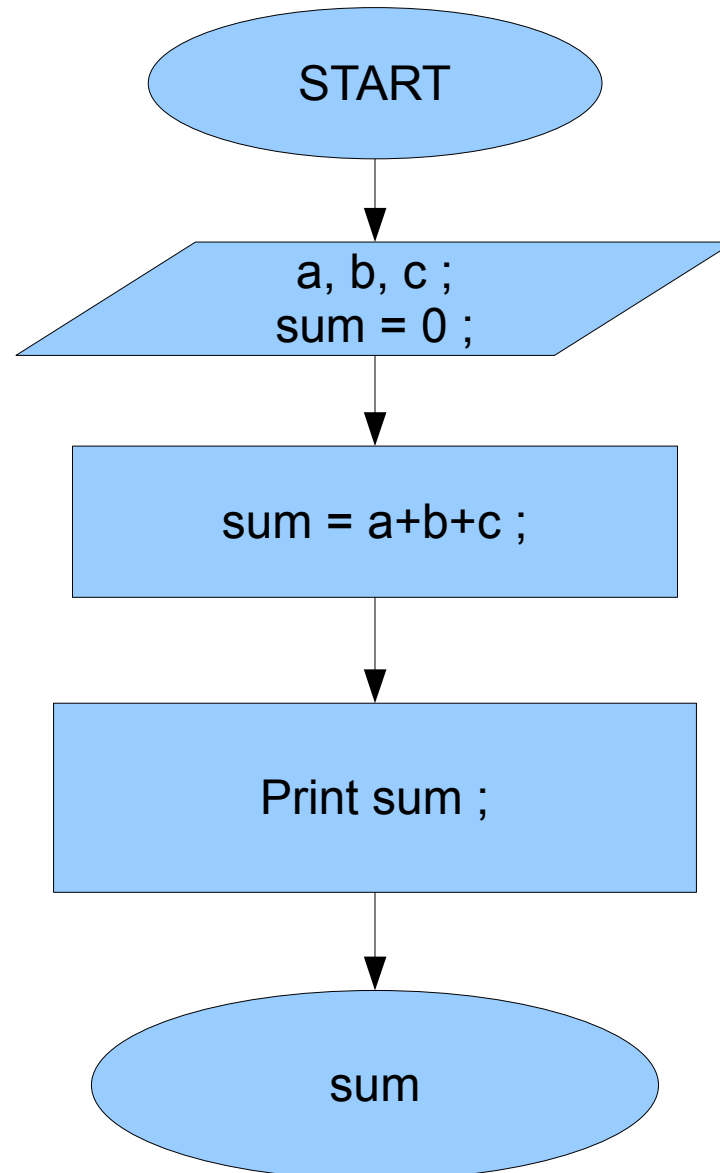
- I/O: Keyboard in, screen out, no loops

Implement the algorithms, cont.

- I/O: Keyboard in, screen out, no loops

```
print "Give a: " ;  
$a = <STDIN> ;  
print "Give b: " ;  
$b = <STDIN> ;  
print "Give c: " ;  
$c = <STDIN> ;  
$sum = $a + $b + $c ;  
print "The sum of a, b and c is $sum." ;
```

The sum of a, b, c



Implement the algorithms, cont.

- I/O: Keyboard in, screen out, while loop (terminates when the evaluated expression is false)

Implement the algorithms, cont.

- I/O: Keyboard in, screen out, while loop (terminates when the evaluated expression is false)

```
$i = 0 ;  
$sum = 0 ;  
while ($i < 5) {  
    print "Give a number: " ;  
    $n = <STDIN> ;  
    $sum = $sum + $n ;  
    $i++ ;  
}  
print "The sum is $sum." ;
```

Implement the algorithms, cont.

- I/O: Keyboard in, screen out, while loop (terminates when the evaluated expression is false)

```
$i = 0 ;
```

```
$sum = 0 ;
```

```
while ($i < 5) {
```

```
    print "Give a number: " ;
```

```
    $n = <STDIN> ;
```

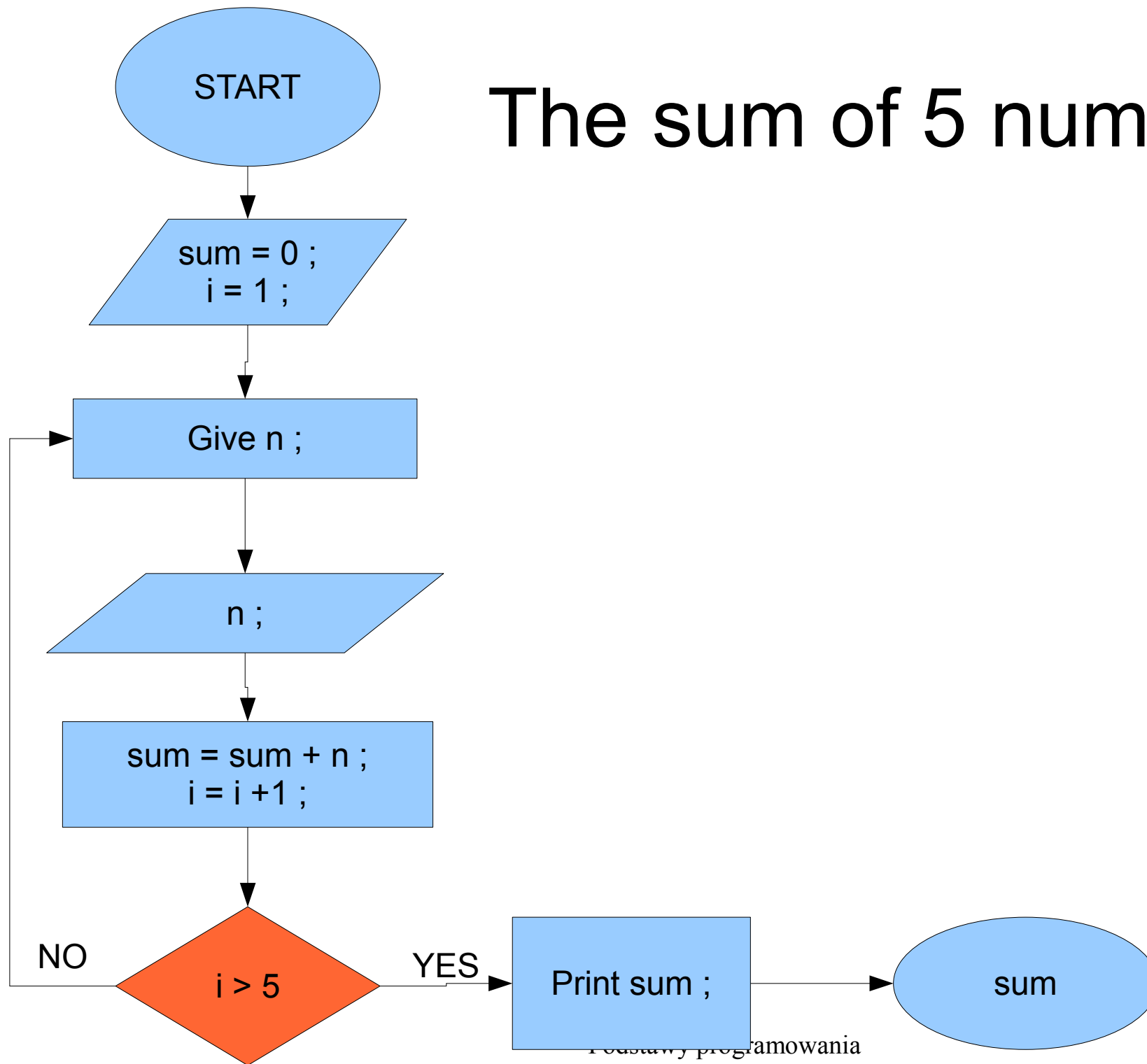
```
    $sum = $sum + $n ;
```

```
    $i++ ;
```

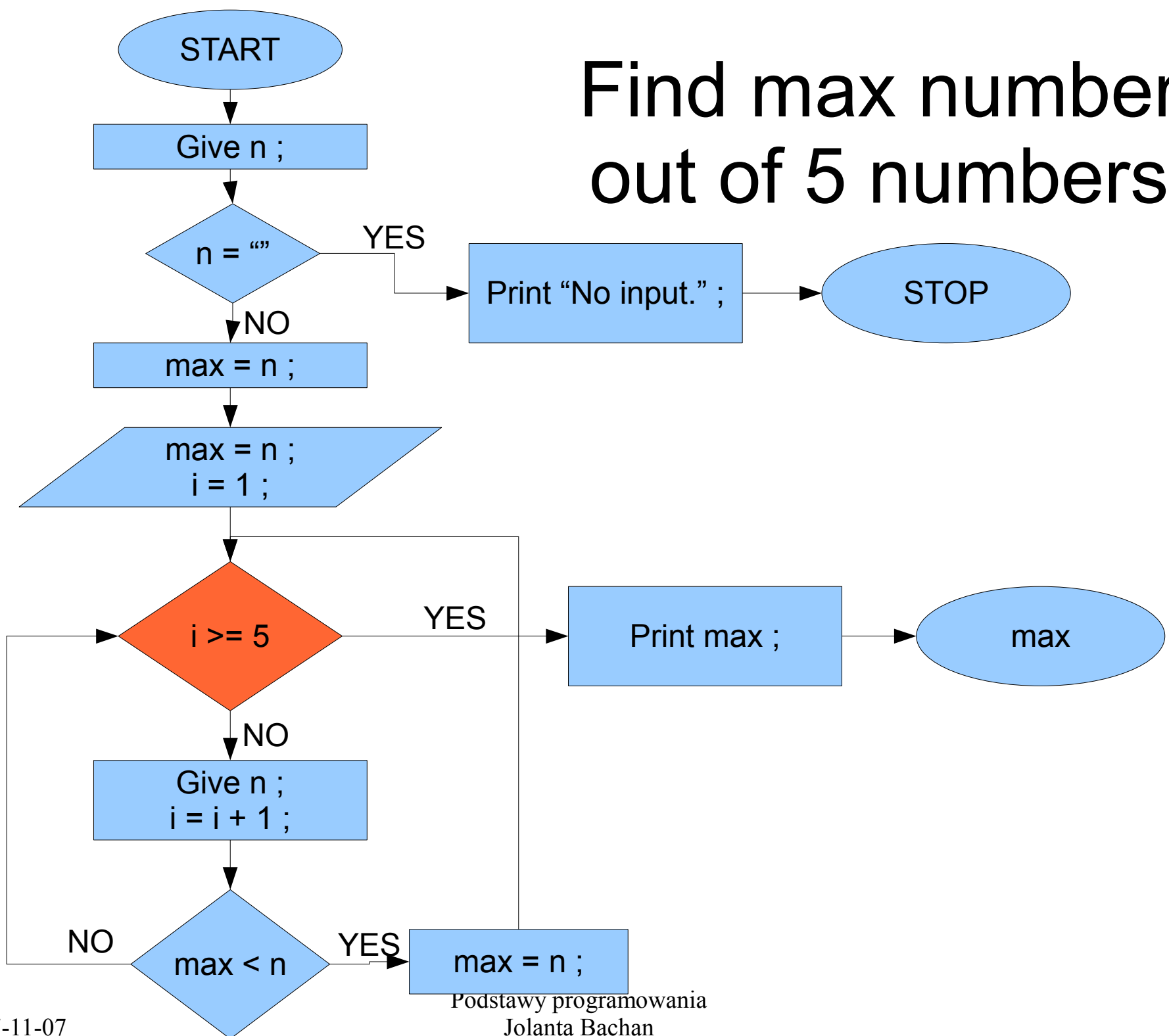
```
}
```

```
print "The sum is $sum." ;
```

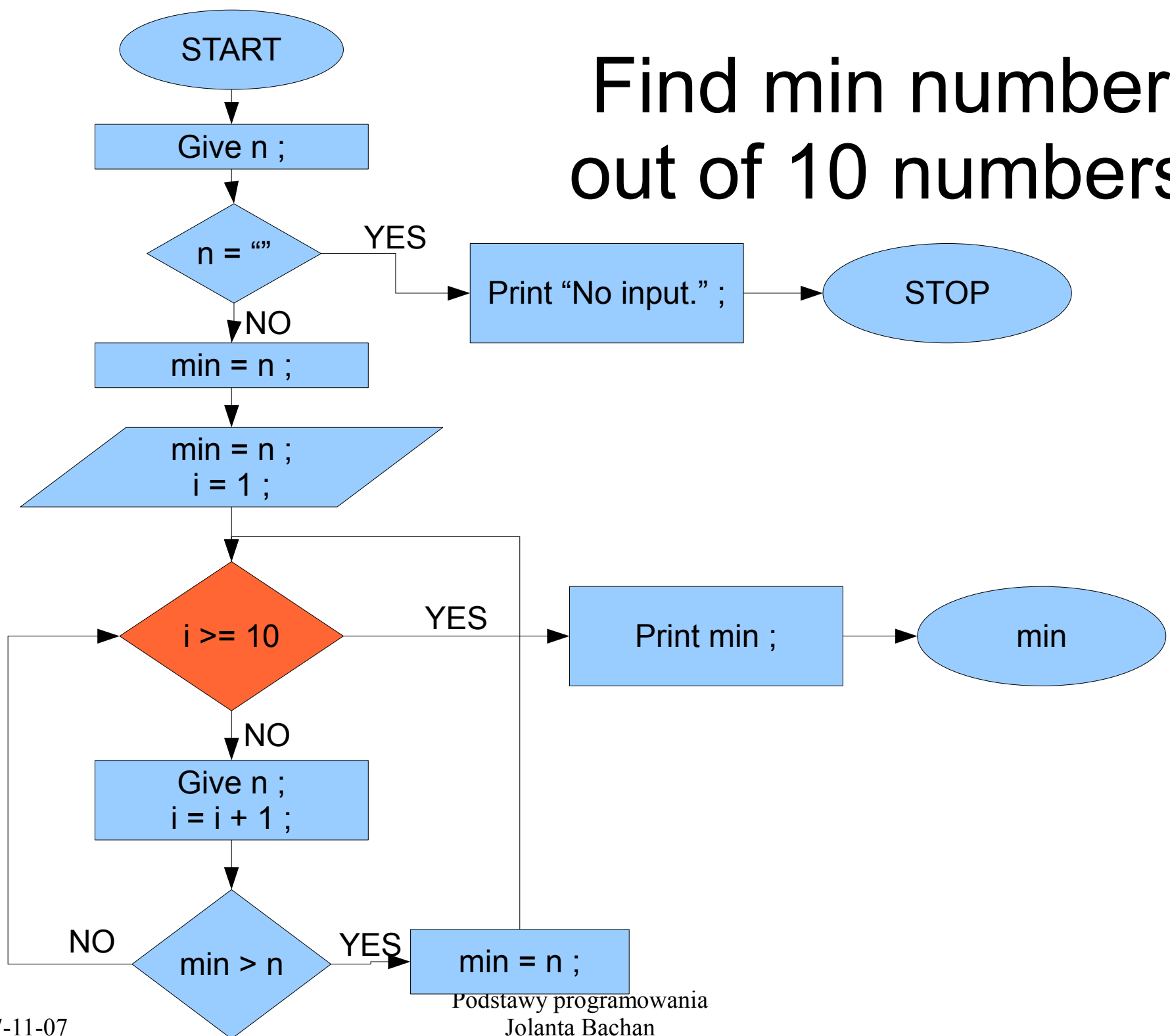
The sum of 5 numbers



Find max number out of 5 numbers



Find min number out of 10 numbers



Output to a file

- I/O: Keyboard in, file out (i.e. file handles, print), while loop

Output to a file

- I/O: Keyboard in, file out (i.e. file handles, print), while loop
- Print OUT function
- Exercise:
 - Napisz program obliczający obwód okręgu. Program ma pobierać wartość promienia od użytkownika programu. (Wzór na obwód wynosi 2π razy promień.)

Output to a file

- I/O: Keyboard in, file out (i.e. file handles, print), while loop
- Print OUT function
- Exercise:
 - Napisz program obliczający obwód okręgu. Program ma pobierać wartość promienia od użytkownika programu. (Wzór na obwód wynosi 2π razy promień.)
 - Następnie drukuj do nowego pliku.

Output to a file

- I/O: Keyboard in, file out (i.e. file handles, print), while loop
- Print OUT function
- Exercise:
 - Napisz program obliczający obwód okręgu. Program ma pobierać wartość promienia od użytkownika programu. (Wzór na obwód wynosi 2π razy promień.)
 - Następnie drukuj do nowego pliku.

```
open (OUT, ">circumference.txt") ;  
print OUT "The circumference is: $circumference\n";
```

Output to a file

- I/O: Keyboard in, file out (i.e. file handles, print), while loop
- Print OUT function
- Exercise:
 - Napisz program obliczający obwód okręgu. Program ma pobierać wartość promienia od użytkownika programu. (Wzór na obwód wynosi 2π razy promień.)
 - Następnie drukuj do nowego pliku.

```
open (OUT, ">circumference.txt") ;  
print OUT "The circumference is: $circumference\n";
```

Append to a file

```
open (OUT, ">>circumference.txt") ;  
print OUT "The circumference is: $circumference\n";  
close (OUT) ;
```

Append to a file

```
open (OUT, ">>circumference.txt") ;  
print OUT "The circumference is: $circumference";  
close (OUT) ;
```

Run the *circumference_part2.pl* and *circumference_part3.pl* programs a few times and check what happens to the *circumference.txt* file.

Close the file

```
open (OUT, ">>circumference.txt") ;  
print OUT "The circumference is: $circumference";  
close (OUT) ;
```

die

- die gets executed only when the result of open is false

```
open (INPUT, "circumference.txt") ||  
  die "Cannot open circumference.txt:  
  $!" ;
```

|| is logical or

\$! is a variable which contains the text relating to the most recent operating system error value, e.g.

```
Cannot open circumference.txt: No such  
file or directory at C:\Documents and  
Settings\aaa\Pulpit\open.pl at line 1.
```

Exercise: *Good2bStudent.pl*

1. Write a program which asks you and your 4 fellow students for a number of hours you and your fellow students study. Save the number of the hours in a file called *StudyHours.txt*. The file should look like:

7

6

8

12

Exercise: *Good2bStudent.pl*

2. Then ask yourself and your fellow students how much money per hour you and they would like to get for studying. Output this information to a file called *MoneyPerHour.txt*.

Exercise: *Good2bStudent.pl*

3. While you ask yourself and your fellow students for the information, the program counts the salary per day and the salary per month (30 days) and outputs this information to *SalaryPerDay.txt* and *SalaryPerMonth.txt* files respectively.

Exercise: *Good2bStudent.pl*

As a result the program creates four TXT files with numbers only:

- StudyHours.txt
- *MoneyPerHour.txt*
- *SalaryPerDay.txt*
- *SalaryPerMonth.txt*

Structure of the program

- Create a loop for 5 iterations
- Ask for the number of hours you study
 - Save in a variable & output to a file
- Ask how much money you would like to get
 - Save in a variable & output to another file
- Count the salary per day and output to a file
- Count the salary per month and output to a file

Input from a file

The Diamond Operator

“<>” is called a diamond operator. It gets its data from the file which is either specified in the command line or in the program.

Exercise: Create a *whilediamond.pl* program.

```
while (<>) {  
    print $_ ;  
}
```

Input from a file

The Diamond Operator

“<>” is called a diamond operator. It gets its data from the file which is either specified in the command line or in the program.

Exercise 1: Write a program with a procedure which opens a file, reads in the data and prints the data to the screen.

```
open (IN, "MoneyPerHour.txt") ;  
while (<IN>) {  
    print $_ ;  
}
```

Input from a file

The Diamond Operator

“ $\langle \rangle$ ” is called a diamond operator. It gets its data from the file which is either specified in the command line or in the program.

Exercise 2: Write a program which opens a file with numbers only, counts the sum and outputs the result to an output file called *sum.txt*.

Back to *Good2bStudent.pl*

Input from a file

Exercise: *Good2bStudent.pl*

4. Open the *SalaryPerDay.txt* file and
 - summarise the figures and save the sum in a `$sum_MoneyPerDay` variable.
 - count the average and save in the result in a `$averagesalary_perDay` variable

Input from a file

Exercise: *Good2bStudent.pl*

5. Open the *SalaryPerMonth.txt* file and
 - summarise the figures and save the sum in a `$sum_MoneyPerMonth` variable.
 - count the average and save the result in a `$averagesalary_perMonth` variable

Output to a file

Exercise: *Good2bStudent.pl*

6. Create a file called *ExpensesAMU.txt* and

– Print the results to this file saying:

```
"For the group of $i students Adam  
Mickiewicz University would have to  
spend $sum_MoneyPerDay Euros every day  
to make them happy. This means that a  
student would earn $averagesalary_perDay  
Euros per day.
```

```
Every month the University would have to  
spend $sum_MoneyPerMonth Euros, which is  
$averagesalary_perMonth Euros for each  
student.
```

```
AMU's students are very demanding."
```

I/O to a file

- Remember to close the opened files when the operations on them are finished!!!

Now chill out!