תוכנה 1

תרגול 1: מערכים ומגנונים בכרת
Useful Eclipse Shortcuts

- **Ctrl+1** – quick fix for errors, or small refactoring suggestions
- **Ctrl+SPACE** – code content assist (auto-completion)
  - Auto completion for “main” – create a template for main function
  - Auto completion for “print” – `system.out.println()`
  - Auto completion for “for” – loop structures
  - And many more, see Window > Preferences > Java > Editor > Content assist > templates
- **Ctrl+Shift+S** – save changes in all open files
- **Ctrl+Shift+F** – auto-formatting of the code (always use it before you submit your HW!)
- **Ctrl+Shift+O** – organize imports (which allows using external classes)
- **Ctrl+F11** – run, **F11** – debug
- **Alt+Shift+R** – rename (a variable, method, class)

All the shortcuts are listed (and can be customized in Window > Preferences > General > Keys)
Array: A fixed-length data structure for storing multiple values of the same type

Example from last week: An array of odd numbers:

Indices (start from 0) → 0 1 2 3 4 5 6 7
odds:

The type of all elements is int
The value of the element at index 4 is 9: odds[4] == 9
Array Variables

- An array is denoted by the [] notation

Examples:

- `int[] odds;`
- `int odds[];`  // legal but discouraged
- `String[][] names;`
- `int[][] matrix;`  // an array of arrays
Array Creation and Initialization

What is the output of the following code:

```java
int[] odds = new int[8];
for (int i = 0; i < odds.length; i++) {
    System.out.print(odds[i] + " ");
    odds[i] = 2 * i + 1;
    System.out.print(odds[i] + " ");
}
```

Output:

0 1 0 3 0 5 0 7 0 9 0 11 0 13 0 15

Array creation: all elements get the default value for their type (0 for int)
Array Creation and Initialization

Creating and initializing small arrays with \textit{a-priori} known values:

- \texttt{int[]} \texttt{odds} = \{1,3,5,7,9,11,13,15\};
- \texttt{String[]} \texttt{months} =
  \begin{verbatim}
  \end{verbatim}
Loop through Arrays

By promoting the array's index:

```java
for (int i = 0; i < months.length; i++) {
    System.out.println(months[i]);
}
```

foreach:

```java
for (String month: months) {
    System.out.println(month);
}
```

The variable month is assigned the next element in each iteration
Operations on arrays

The class Arrays provide operations on array

- Copy
- Sort
- Search
- Fill
- ...

[java.util.Arrays](http://docs.oracle.com/javase/6/docs/api/index.html?java/util/Arrays.html)
Copying Arrays

- Assume:
  int[] array1 = {1,2,3};
  int[] array2 = {8,7,6,5};

- Naïve copy:
  array1 = array2;

- What’s wrong with this solution?
Copying Arrays

- **Arrays.copyOf**
  - the original array
  - the length of the copy

```java
int[] arr1 = {1, 2, 3};
int[] arr2 = Arrays.copyOf(arr1, arr1.length);
```

- **Arrays.copyOfRange**
  - the original array
  - initial index of the range to be copied, inclusive
  - final index of the range to be copied, exclusive
What is the output of the following code:

```java
int[] odds = {1, 3, 5, 7, 9, 11, 13, 15};
int newOdds[] = Arrays.copyOfRange(odds, 1, odds.length);
for (int odd: newOdds) {
    System.out.print(odd + " ");
}
```

Output: 3 5 7 9 11 13 15
2D Arrays

- There are no 2D arrays in Java but …
- you can build array of arrays:
  ```java
  char[][] board = new char[3][];
  for (int i = 0; i < 3; i++)
    board[i] = new char[3];
  ```

Or equivalently:
```java
char[][] board = new char[3][3];
```
2D Arrays

A more compact table:

```java
int[][] table = new int[10][];
for (int i = 0; i < 10; i++) {
    table[i] = new int[i + 1];
    for (int j = 0; j <= i; j++) {
        table[i][j] = (i + 1) * (j + 1);
    }
}
```

Table:

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>9</td>
</tr>
</tbody>
</table>
```
Fibonacci

- Fibonacci series
  1, 1, 2, 3, 5, 8, 13, 21, 34
- Definition:
  - fib(0) = 1
  - fib(1) = 1
  - fib(n) = fib(n-1) + fib(n-2)

en.wikipedia.org/wiki/Fibonacci_number
public class Fibonacci {

    /** Returns the n-th Fibonacci element */
    public static int computeElement(int n) {
        if (n==0) {
            return 1;
        } else if (n==1) {
            return 1;
        } else {
            return computeElement(n-1) + computeElement(n-2);
        }
    }
}

Assumption: 
\[ n \geq 0 \]
public class Fibonacci {

/** Returns the n-th Fibonacci element */
public static int computeElement(int n) {
    switch(n) {
        case 0:
            return 1;
        case 1:
            return 1;
            break;
        default:
            return computeElement(n-1) + computeElement(n-2);
    }
}
}

/* Compilation Error: Unreachable Code */

Assumption: n≥0
Iterative Fibonacci

A loop instead of a recursion

```java
static int computeElement(int n) {
    if (n == 0 || n == 1)
        return 1;
    int prev = 1;
    int prevPrev = 1;
    int curr;
    for (int i = 2; i < n; i++) {
        curr = prev + prevPrev;
        prevPrev = prev;
        prev = curr;
    }
    curr = prev + prevPrev;
    return curr;
}
```

Assumption: 

\[ n \geq 0 \]
נתונים במיקום חישוב

בתרגום רקורסיה לולאות אנו משתמשים במשתנים prevPrev - prev curr, prev
😻 בין לשירות המчувствת הלולאה "זוכרת" את הנקודת שבבה אנו נמצאים
בתהליך החישוב

דינו:עילות לעומת פשטונת
(keep it simple stupid) KISS

עריגות: כתבו את השירות
computeElement

תרוגת: כתבו את השירות
(curr Lal prevPrev - prev curr Lal prev

ב userDao (לאו)
For Loop

Printing the first n elements:

```java
public class Fibonacci {
    public static int computeElement(int n) {
        ...
    }

    public static void main(String[] args) {
        for(int i = 0 ; i < 10 ; i++) {
            System.out.println(computeElement(i));
        }
    }
}
```

It is better to use args[0]
מודולים, שכפול קוד ויעילות

יש כאן חוסר יעילות מסוים:

- לולאת ה-for קוזרת גמ-
- לולאת main גמ-
- לכרות, במעבר אחד
- ניטGREEK
computeElement גמ-
- ניטGREEK
- גבע

כמו כל可以根据 איבר בסדרה איננו ו kontakt
- במעבר שכרby
- הבilestone של איברים קוזר
- ידוע

ומ تعالى כל יישוב מתחלתו
מודולים, שכפול קוד ויעילות

מתודה (פונקציה) זריכה לעשות דבח אחד בדיק!
עורב של חישוב והדפסה פוגע במודולים, והזהר לשכפול קוד!
קטע קוד דומה המופיעה בשתי פונקציות תשומת יגום
במוקדם או במאוחרifferences בין בתוכנites (מודע)
את בעיות היעילות (חוספת מנוגון חום)
apesh לفاتור בעריך מערכיס (תרגיל)
The following two statements are almost equivalent:

```java
for(int i = 0 ; i < n ; i++)
    System.out.println(computeElement(i));
```

```java
int i = 0;
while (i < n) {
    System.out.println(computeElement(i));
    i++;
}
```

Variable `i` is not defined outside the for block.
while vs. do while

The following two statements are equivalent if and only if $n > 0$:

```java
int i=0;
while (i < n) {
    System.out.println(computeElement(i));
    i++;
}

int i=0;
do {
    System.out.println(computeElement(i));
    i++;
} while (i<n);
```
Compilation vs. Runtime Errors

Compilation Errors (Cúmpling): Errors that can be "caught" at the time of reading the file and converting it to bytecode.

Examples:

```java
Class MyClass {
    void f() {
        int n = 10;
    }
    void g() {
        int m = 20;
    }
}
```

Syntax error on token "Class", class expected

```java
int i;
System.out.println(i);
```

Syntax error, insert "}" to complete MethodBody

```
main(String[] args) {
    int i;
    System.out.println(i);
}
```

The local variable i may not have been initialized
1 quick fix available:
- Initialize variable

Errors are common:
- Syntax
- Type fitting
- Defining before use

בדרק כל קשרות:
- תוחבר תארו תיפוסים, הגדרת לאפי שימו
Compilation vs. Runtime Errors

Compilation Errors:

- Cannot be known at compile time:

  - at compile time (компилирование)

Examples:

- Exception handling (exceptions), which we’ll learn later.

```
int a[] = new int[10];
...
a[15] = 10;
...
String s = null;
System.out.println(s.length());
...
```

Runtime Errors:

- Exception handling (exceptions), which we’ll learn later.

```
public class Main {
  public static void main(String[] args) {
    String s = null;
    System.out.println(s.length());
  }
}
```

```
Exception in thread "main" java.lang.NullPointerException
  at Main.main(Main.java:4)
```
Compilation vs. Runtime Errors

אהמيشעוזסומDefaultValue?

כן,הכיגורשותטעויותלוגיותבתוכנית

```java
public class Factorial {
    /** calculate x! **/
    public static int factorial(int x) {
        int f = 0;
        for (int i = 2; i <= x; i++)
            f = f * i;
        return f;
    }
}
```
The Debugger

- Some programs may compile correctly, yet not produce the desirable results.
- These programs are valid and correct Java programs, yet not the programs we meant to write!
- The debugger can be used to follow the program step by step and may help detecting bugs in an already compiled program.
Debugger – Add Breakpoint

• Right click on the desired line
• “Toggle Breakpoint”
Debugger – Start Debugging

test.java

public class Test {

    public static void main (String [] args){
        System.out.println(computeFibElement(7));
    }

    public static int computeFibElement(int n) {
        if (n == 0 || n == 1)
            return 1;
        int prev = 1;
        int prevPrev = 1;
        int curr;
        for (int i = 2 ; i < n ; i++) {
            curr = prev + prevPrev;
            prevPrev = prev;
            prev = curr;
        }
        curr = prev + prevPrev;
        return curr;
    }
}
Debugger – Debug Perspective

This kind of launch is configured to open the Debug perspective when it suspends.

This Debug perspective is designed to support application debugging. It incorporates views for displaying the debug stack, variables and breakpoint management.

Do you want to open this perspective now?
Debugger – Debugging

Current state

Current location

Back to Java perspective
Debugger – Debugging
Using the Debugger: Video Tutorial

מצגות וידאו
http://eclipsetutorial.sourceforge.net/debugger.html

מדריך עדכני יותר
http://www.vogella.com/tutorials/EclipseDebugging/article.html

הקישורים נמצאים גם באתר הקורס
...ทราบ