2XB3:
Practice and Experience: Binding Theory to Practice

Lecture 1 - Introduction

Reza Samavi

Outline

- 2XB3 Course overview
- Integrated Development Environment (IDE)

2XB3:
Practice and Experience: Binding Theory to Practice

Instructor

Reza Samavi, PhD, PEng
Email: samavir@mcmaster.ca

Course Website

A2L: https://avenue.cllmcmaster.ca/

About me: Reza Samavi

- PhD in Information Engineering from University of Toronto
- Affiliated with the Vector Institute for AI
- Professional Engineer in the province of Ontario
- 11 years work experience in industry
  - co-founder and CEO of a software development start-up
- Research
  - information security, artificial intelligence, machine learning, semantic web

VECTOR INSTITUTE | INSTITUT VECTEUR

SPAI: Security, Privacy & AI Research Group at McMaster

McMaster University

Security, Privacy & AI (SPAI) Research Group at McMaster

- Faculty member: **Reza Samavi**, PhD, PEng
- Faculty of Engineering, Department of Computing and Software
- Affiliated with the Vector Institute for AI
- eHealth Graduate Program Lead
- Current members: 9 graduate students (3 PhD)

SPAI@McMaster Research Group

Security	Machine Learning	Semantic Web

Transparency and accountability algorithms
Algorithmic security of Machine Learning
Health analytics & semantics

Funding

NSERC CRSNG

SSCIP SMART COMPUTING FOR INNOVATION

Collaborators

IBM

University of Toronto

Autodesk Research

Parametric Human Project

Hamilton Health Sciences

Rouge Valley Health System

Course objectives

- Allow students to apply computational solutions, learned in theories, on practical problems
  - theoretical analysis learned in **Data Structure and Algorithms** (2C03)
  - development techniques acquired through **software design methods** (2ME3/2AA4)
- Facilitate **experiential learning** through individual and team-work
- Learn to improve your inter-personal relationships within the team to implement a software project

By successfully completing this course you should be able to…

- implement in Java fundamental and broadly useful data structures and user defined ADTs.
- implement in Java, basic searching, sorting, and graph algorithms.
- implement in Java basic algorithm cost models and run experiments in Java to analyze running times of different algorithms.
- formulate and implement test plans using JUnit.

- explore and formulate a programming problem (represented as an open engineering problem) that requires algorithmic solutions
- working in a software development team and play the role of a software engineer to implement small data-intensive software projects with algorithmic contents.
- Use version control systems to coordinate work of multiple programmer in a team.
- evaluate a software development team dynamics.

Top 8 Technology Trends for 2020

1. Artificial Intelligence (AI)
   - navigation apps, streaming services, smartphone personal assistants, ride-sharing apps, home personal assistants, and smart home devices.
   - schedule trains, assess business risk, predict maintenance, and improve energy efficiency
   - jobs in AI will number 23 million by 2020
2. Machine Learning
   - Machine Learning is a subset of AI, we also have subsets within the domain of Machine Learning, including neural networks, natural language processing (NLP), and deep learning.
   - Rank among the top emerging jobs on LinkedIn

Source: https://www.simplilearn.com/top-technology-trends-and-jobs-article

Top 8 Technology Trends for 2020

3. Robotic Process Automation or RPA
4. Edge Computing
5. Virtual Reality and Augmented Reality
6. Blockchain
7. Internet of Things (IoT)
8. Cybersecurity

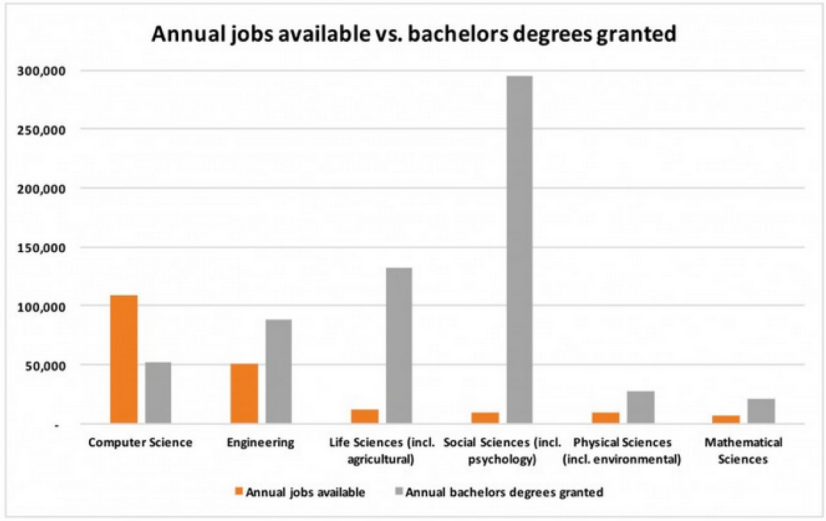
Source: https://www.simplilearn.com/top-technology-trends-and-jobs-article

Where the jobs are – 2016 edition

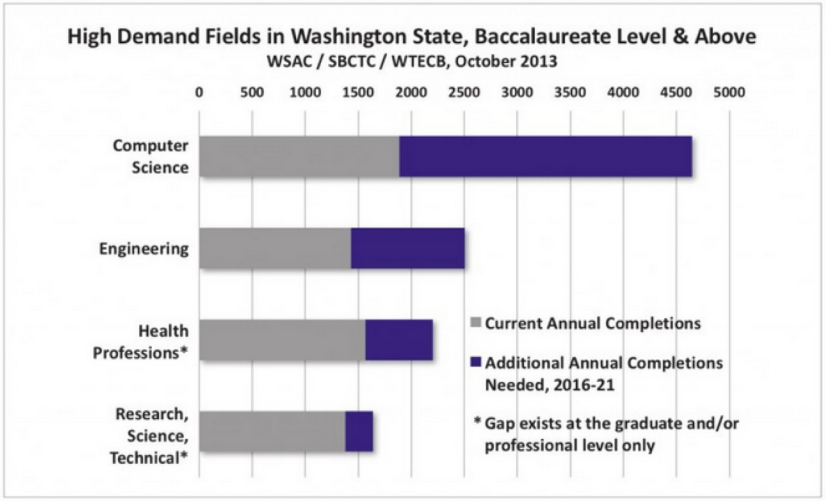
The 2016 NSF Science & Engineering Indicators have recently been released, including data on the numbers of degrees granted in various fields of science and engineering.

Similarly, the 2016 US Bureau of Labor Statistics job projections have recently been released, covering the decade 2014-2024.

The Tale of the Tape: A chart of "annualized jobs available" (from BLS) vs. "annual degrees granted" (from NSF) for various fields.



The story in Washington State is the same. Three state educational agencies looked across all fields (not just STEM fields) for those with any appreciable gap between "degrees granted" and "jobs available." They found only four at the bachelors or graduate level. Computer science was #1, with a gap 2.5X as great as the second-place field – which was all fields of engineering added together (electrical, mechanical, civil, aeronautical, materials, etc.). (Of course, not all of those computer science jobs are in Washington's vibrant software industry. For example, Washington's aerospace industry employs roughly 3X as many computer scientists as aeronautical engineers!)



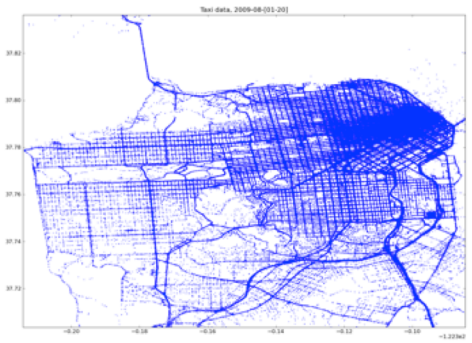
Wowsers!

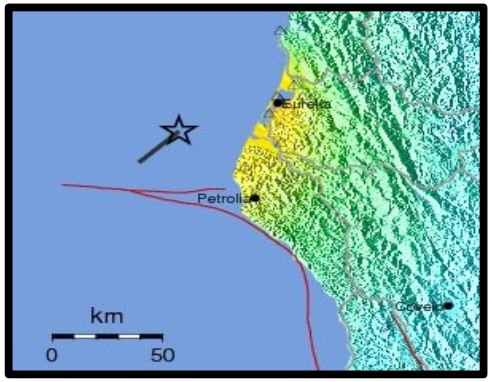
March 27, 2016

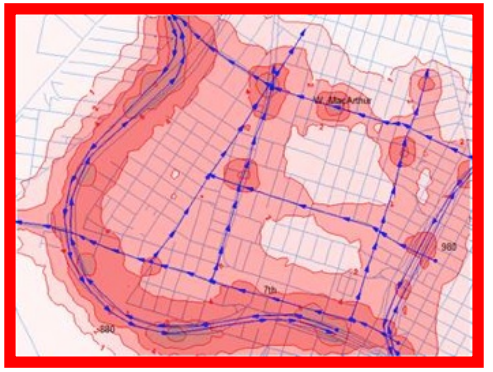
Processing (Big!) Data is at the core of this course

- Why?

What can you do with the data?
From Traffic Prediction to Earthquake Warning







Crowdsourcing + physical modeling + sensing + data assimilation

to produce:

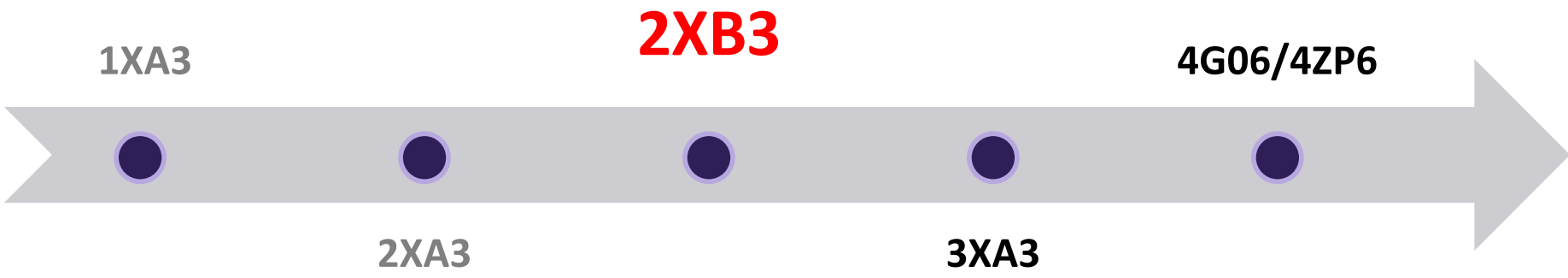
Processing (Big!) Data is at the core of this course

- Why?
- The advertised salary for technical professionals with big data expertise and in-demand skills is $124,000 net of bonuses and compensation.
- IBM, Cisco and Oracle together advertised 26,488 open positions that required big data expertise in the last twelve months.
- EMC (Dell) has 25.1% of all available big data positions that WANTED Analytics tracks.

Source: http://www.forbes.com/sites/louiscolumbus/2015/11/16/where-big-data-jobs-will-be-in-2016/#673e86d6f7f1

Series of experiential learning courses

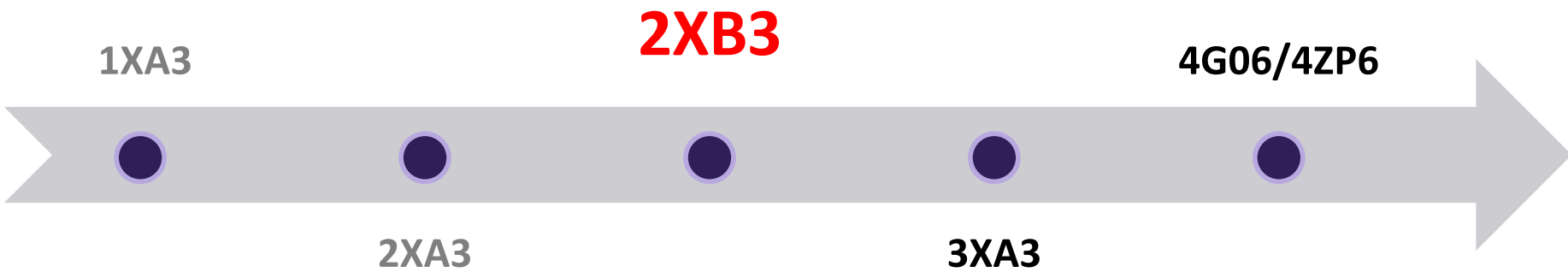
- learning through reflection on doing
  - Application of computational theories
  - Implementation and programming
  - Progressing from programming to implementing information systems
  - Progressing from individual work to team work
- Preparing you to do an independent capstone project



Experiential learning courses

- learning through reflection on doing
- **SE/CS 2XB3**: Practice and Experience: Binding Theory to Practice

  Open-ended design of computational solutions to practical problems that involve both theoretical (algorithmic) analysis and implementation; solving computational problems through an experiential approach; revision and version control.



2XB3: learning through reflection on doing

- Start from application of computational theories
- Start from implementation and programming as individual work
- Progress to implementation information systems
- Progress to team work

Machine Learning (ML)
High performance computing (HPC)

Data anayltics plat

Web applications with algorithmi

Software Engineering

Software Versioning **(SVN)**

Program Design **(MVC and JDBC)**

Documentation **(JavaDoc)**

Software Development Methodology planning, and monitoring (**Agile methods: Scrum**)

Programming

ADT

Experiments

Testing

Sorting

Searching

Graphs

Final Project

Team

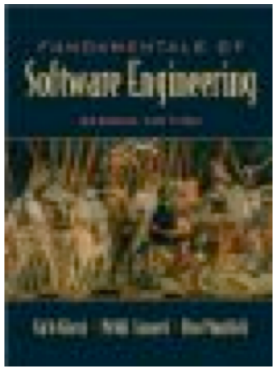
Assignments & in Lab projects

Individual

References

Sedgewick, R., and K. Wayne. "Algorithms (4the Edition).

"Fundamentals of Software Engineering", Carlo Ghezzi, Mehdi Jazayeri, Dino Mandrioli Second Edition



- **Java**

  Thinking in Java, 3rd Edition" (purchasing the latest edition is recommended if you are interested in a paper copy).

  http://www.mindview.net/Books/TIJ/

  Oracle online tutorials:

  http://docs.oracle.com/javase/tutorial/

  Head First Java, 2nd Edition, O'Reilly Media

Java Programming

- The free electronic edition of the book "Thinking in Java, 4th Edition" or purchasing the latest edition is recommended if you are interested in a paper copy.
- This reference is the recommended background reading material for Java, to be used as a refresher source for the prerequisite Java information that is required by this course. There are many other online and paper copy Java references available that you can use, such as:
- Oracle online tutorials:

http://docs.oracle.com/javase/tutorial/

- Head First Java, 2nd Edition, O'Reilly Media

Grading and Assessment

Assignments	30%
Java quiz and Lab participation	30%
Final Project (team)	40%

Lecture Participation

- Lecture participation is not mandatory but there is a bonus mark up to 2% of the final mark
  - 2% for at least 80% participation, 1.5% for at least 70% participation, 1% for at least 60% participation.
- i>clicker is used for counting your participation
- Your performance will not be measured but your participation

Labs

- There are four hours of lab per week per section
- Walk-through labs
  - Students will be walked through by the TA on the implementation of the algorithmic contents.
  - You should submit your completed example.
  - Practice problems will be provided.
- Practice labs - Java Tutorial Labs
  - Provide opportunity for self-guided work
  - TAs will be in the labs to answer your questions
  - Mandatory team meetings will occur during the first 45 minutes of practice labs
  - Cross-team meetings may occur during the practice labs
- Lab participation is mandatory and count for 5% of the final mark
  - 5% for attending at least 90% of the labs, 4% for at least 80%, 3% for at least 70%, 2% for at least 60%, and 1% for at least 50%
  - No need to MSAF a missing walk-through or practice lab as you can get the full mark even if you miss one of each

Quizzes

- There are four quizzes (mini problems)
- First quiz on Java programming
  - You need self study to improve your Java programming skill
  - The quiz is worth 6%
- Second quiz on ADT & Sorting
  - The quiz is worth 6%
- Third quiz on Sorting & Searching
  - The quiz is worth 6%
- Fourth quiz on Searching & Graph
  - The quiz is worth 7%
- If you MSAF a quiz the weight will be shifted to the next remaining quizzes except for the last quiz. If you MSAF the last quiz, there will be a make up quiz to take.

Assignments

- There are two homework assignments (A1:15%, A2:15%)
- You have usually two weeks to complete each assignment
- Requires Java implementation of algorithmic solutions for open engineering problems, i.e. you may require to make reasonable assumptions
- Using external libraries are allowed with the following conditions
  - Acknowledge and explicitly give proper credit with meaningful comments inside your code
  - Properly cited external codes can only be included as Java libraries i.e. you are not allowed to copy full or partial codes from other resources and include them inside your code
  - The included libraries should not be a substantial part of your assignment
- Grading
  - A submitted solution that does not compile or run gets 0 credit
  - A solution that runs but is partially correct gets partial credit
  - Providing adequate, concise, and meaningful comments throughout your code is part of the solution grade
- Every hour late will cost 2% of the assignment mark. After 48 hours the assignment will no longer be accepted and the student will get 0 credit
- All assignments deemed to be substantially similar to each other will get 0 credit
- In order to graduate from this course, all assignments must be completed.

Final Project

- The final project is an important component of this experiential course.
- The main learning objective of the final project is to prepare you for developing software for real world problems.
- The final project is a software implementation project with algorithmic content.
- By doing the final project, you will experience team-work in a situation similar to the real world software development environment.

- Different steps:
  - Forming a team (due by the end of this week)
  - Pre-study
  - Project proposal and its presentation
  - Requirements specifications
  - Design specifications
  - Final project implementation
  - Final project presentation
  - Team peer evaluation (individually completed)

Please read the Final Project PDF document on the course website for a detailed description

Final Project – Forming a team

- Early in the semester, students will form teams
- All students in a team must belong to the same lab section.

Final Project – Forming a team

Final Project – Forming a team

Final Project - Project Proposal

- Every team will propose a software application
  - Must require algorithmic content in the application level
    - a web service that finds the shortest path between two points,
    - a web application that schedule final exams in a department,
    - a mobile application that remove repetitive pictures from the photo gallery
  - Makes use of one of the publicly available Canada or US Government open datasets
    - links are available in the project document
    - dataset size must exceed 10MB
  - Makes use of sorting, searching, and graph algorithms
  - Can be implemented in about 8 weeks by the team
- The document is limited to 2 pages and must be completed using the project proposal template

Final Project - Project Proposal

- Every student will propose a software application
  - Must require algorithmic content in the application level
    - a web service that finds the shortest path between two points,
    - a web application that schedule final exams in a department,
    - a mobile application that remove repetitive pictures from the photo gallery
  - Makes use of one of the publicly available Canada or US Government open datasets
    - links are available in the project doc
    - dataset size must exceed 10MB
  - Makes use of sorting, searching, ar
  - Can be implemented in about 8 we
  - The application is approved by the
- The document is limited to 2 pages and must be completed using the project proposal template

start early!

Final Project – Forming a team

Final Project – Forming a team

Final Project – Project Proposal Presentation

- Each team has 10 minutes to present the proposal
- Other students will provide feedback on their peer's proposal
  - The feedback and comments will not impact your grade but will help others in a team to pick a topic to work on

Final Project – Project Implementation

- Project planning
  - You may consult the feedbacks from your peers' to scope the project better
  - You cannot switch to a new topic, but you are allowed to narrow the scope
- Identify the requirements
- Create a design concept
- Implement it as a prototype
- You will provide the following deliverable
  - Requirements specifications
  - Design specifications
  - Implemented project
- Finally the work carried out in the project will be presented by all members of the teams to the class

Final Project - Peer Review

- At the end of the project, every team member must individually submit a review of other team members.
- For each person reviewed, the review should specify
  - the amount of interaction that the reviewer had with the person
  - an evaluation of the following three criteria
    - The effort that the person put into the project.
    - The quality of the work performed.
    - The person's professionalism in terms of meeting deadlines, doing their share of the project, being easy (and even pleasant) to work with, etc.
- You will assign a score for each criterion.
- You will write comments, justifying the scores given.
- The reviews are independent and confidential
- The peer evaluation will be factored into students combined grade for the team work.

Email Policy

- Email is the preferred way to contact the instructor and the teaching assistants
- Please use descriptive subject lines for your emails and include “SE 2XB3”, “CS 2XB3” at the beginning of the subject
  - e.g. “SE 2XB3 - prerequisite waiver request”
- Please sign your email with your full name and student #
- Please send email only from your official McMaster University email address
- We will try to answer your emails in 1 working day. You should not rely on getting same-day answers (particularly near assignment deadlines)

Outline

- 2XB3 Course Overview
- Integrated Development Environment (IDE)

About IDEs

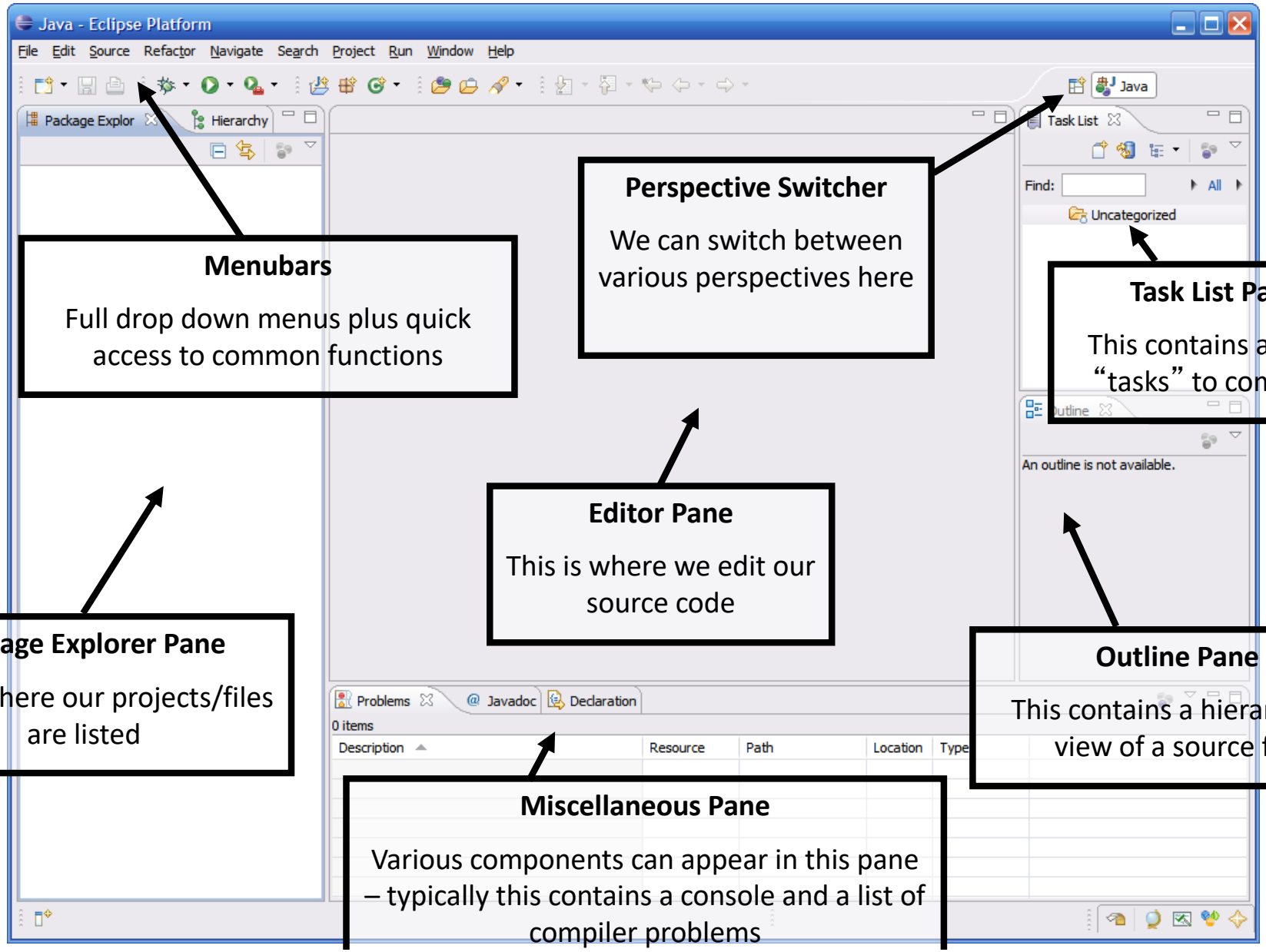
- An IDE is an Integrated Development Environment
- Different IDEs meet different needs
  - BlueJ, DrJava are designed as teaching tools
    - Emphasis is on ease of use for beginners
    - Little to learn, so students can concentrate on learning Java
  - Eclipse, JBuilder, NetBeans are designed as professional-level work tools
    - Emphasis is on supporting professional programmers
    - More to learn, but well worth it in the long run
- For this course we will use Eclipse, but other professional IDEs are similar

What is Eclipse?

- Eclipse started as a proprietary IBM product (IBM Visual age for Smalltalk/Java)
  - Embracing the open source model IBM opened the product up
- Open Source
  - It is a general purpose open platform that facilitates and encourages the development of third party plug-ins
- Best known as an Integrated Development Environment (IDE)
  - Provides tools for coding, building, running and debugging applications
- Originally designed for Java, now supports many other languages
  - Good support for C, C++
  - Python, PHP, Ruby, etc…

Slide from: www.eclipse.org/eclipse/presentation/eclipse-slides.ppt

Eclipse IDE Components



Menubars

Full drop down menus plus quick access to common functions

Perspective Switcher

We can switch between various perspectives here

Task List Pane

This contains a list of "tasks" to complete

Editor Pane

This is where we edit our source code

Package Explorer Pane

This is where our projects/files are listed

Outline Pane

This contains a hierarchical view of a source file

Miscellaneous Pane

Various components can appear in this pane – typically this contains a console and a list of compiler problems

What you learn about Eclipse in the lab?

- Creating a Workspaces / Projects
- Creating Classes
- Compiling and Running Code
- Debugging Code
- Sampling of Features