Introduction to Course

Course description

The *University of Manitoba Undergraduate Calendar* describes this course as follows:

> *An introduction to computer programming using a procedural high level language.*
> *Not to be held with COMP 1011 or the former 074.112, 074.121, 074.123, or 074.125.*

This course assumes no prior knowledge of programming. The course does assume that the student is familiar with using a computer for tasks such as browsing the web, basic use of application software, and installing new software. The intent is to learn problem solving and computer programming, for a variety of students, using a high level computer programming language. Currently, the programming language is Java (which is an high-level object-oriented programming language).

Course goals

Upon completion of this course, you will be able to:

- summarise the fundamentals of computers;
- explain and follow the process of writing, compiling, and executing programs in high level programming languages, and infer how Java's "write once, run anywhere" strategy alters this process;
- distinguish between different primitive data types and where they should be used;
- identify the differences between primitive and complex (class) data types;
- write complicated mathematical expressions using operators and methods;
- utilise program input and output;
- create programs that use decision-making and repetition structures;
- develop step-by-step methods for solving elementary problems, and use these methods to write appropriate computer programs;
- apply a strategy to reduce more complicated problems to a sequence of methods;
- analyse simple programs to establish what the output and behaviour would be;
- locate, identify, and fix errors in programs;
- construct and use simple classes in object-oriented programming; and
- apply arrays to store and manipulate collections of data.

Course materials

Computer system requirements

A programming platform/environment on which a recent web browser (one on which you can view this site) and Java can run (e.g., Windows XP or newer for Intel machines). At least a Pentium processor is required on those machines. If you have access to a Macintosh (running OS X), or a computer running UNIX or Linux, you may use them, but the Distance and Online Education Program will not be able to provide support.

If you are close to the University of Manitoba Fort Garry campus, you may use the computers running a Windows operating systems in the open area computer labs. See the ACN web site for details.

Textbook

The official textbook for this course is *Starting Out with Java: From Control Structures through Objects* 5th ed. by Tony Gaddis, Addison Wesley, 2012. This textbook which is also being used by online COMP 1020 (the follow-up course to COMP 1010) and the in-person sections of COMP 1010 and COMP 1020. The book may be purchased from any source, but it is available in the University of Manitoba bookstore. There is a companion web site for the book.

While the current version of the textbook is the most up to date and will therefore be the most helpful as a reference in future studies, you may use older editions of the textbook for this course if you have a copy (4th ed. link, 3rd ed. link). Keep in mind that chapter/section numbers and practice problems are not guaranteed to be the same in older editions. Since older editions are not listed as required, the University of Manitoba Bookstore will not stock copies for sale.

Software

No software needs to be purchased for this course, other than what you are already using to view this web page. However, some software packages must be downloaded and installed to your computer.

The Java Software Development Kit (JavaSDK) and a text editor such as Textpad are both required. Downloading this software is part of Assignment 0. If you wish to download your software now, please visit the assignments link from the home page, select Assignment 0, and follow the instructions for Activity 1.

Documentation

The document Programming Standards for COMP 1010 is required reading. These programming standards must be followed in all assignments submitted for this course.

Additional reading (optional)

The external links page includes a list of links to web sites with additional information that you may find useful.

Backup media

Disks (such as recordable CDs, DVDs, or Zip disks) for making backup copies of your assignments.

Course overview

This course will be divided into ten units, numbered 0 through 9, covering the following topics:

0. **Fundamentals of programming**
   An overview of computers and programming languages; how to code (i.e., write in a computer language) a computer program.
   Programming environments
   Discussion of the programming environment that will be used as the medium of instruction for this course, and how to run a program in that environment
1. **Data types, statements, and simple expressions**
   Primitive data types for storing numbers; statements and expressions for processing primitive data. Floating-point arithmetic. Use of character strings.
   Input and Output
   Input and output of data using the above types.
2. **Advanced expressions**
   Expressions using methods for advanced mathematical and string processing.
   Boolean expressions
   Boolean expressions using the relational and logical operators.
   Control structures: decision-making
   The use of control structures for decision making. Nested control structures and the use of blocks.
3. **Control structures: repetition**
   The use of control structures for repetition (loops). The major varieties of loops, and nested loops.
4. **Problem decomposition: methods**
   Using static methods to decompose complicated problems into more manageable steps, and to separate out repeated code.
5. **Programming style**
   Using javadoc to document methods; techniques of source code style and indentation.
   Debugging
   Techniques for tracking down run-time errors.
6. **Problem decomposition: classes**
   The use of simple worker classes and methods. Constructing objects of classes specified by the programmer.
7. **More classes**
   Types of methods, including accessors, mutators, and constructors. More complex uses of classes. References to objects.
8. **Introduction to arrays**
   The use of arrays to store and process homogeneous collections of data.
9. **Array algorithms**
   Implementing common array algorithms as methods.

There is some overlap between these topics, meaning some topics will extend into other units. This outline should be considered a rough guide; a more detailed outline is presented with each unit.

Each of these topics will be covered in the form of a week-long unit. At the end of each unit, you will be tested on your knowledge of the unit with an assignment and a quiz. Since you will require some time to complete each assignment,

prepare to start each unit **at least two weeks** before the assignment is due. This will allow you to work on the assignment while reading ahead for the next unit (much like you would in an on-campus course, where you would continue to attend lectures even while working on an assignment). The course schedule will be posted early in the term in the form of assignment and quiz due dates; take note of those dates and plan your own work schedule accordingly.

Course Notes

Part of the required work for each unit will be to read the course notes. They provide discussion of the course material independent of the textbook. Some topics may be covered in both sources, some in only one or the other. You are responsible for learning *all* of the material in the course, whether it is from the notes or the textbook.

To make the notes easier to follow, certain typographical conventions have been used:

- ***Definitions*** — words appearing in this format are terms defined in the course glossary.
- *References* — the names of portions of the textbook or this web site.
- `Code` — brief examples, keywords, or identifiers in the Java programming language.
- *`Output`* — the results produced by programs.
- `Filenames` — the names (or portions of names) of files saved on a computer.

Longer portions of sample Java programs, or complete sample programs, are shown as follows:

```
public class Greetings {
  public static void main(String[] args) {
    // Say hello to everybody out there
    System.out.println("Hello!");

    // The following means that some part of the program has been omitted
    // (it's not relevant to the example):
    // ...

    System.out.println("Goodbye.");
  }
}
```

When these examples are complete programs (something you will learn to recognise soon), it is **highly recommended** that you try them out by copying the program text from your web browser, pasting it into your text editor, and running the program. You will learn how to edit and run programs in the first unit of the course.

Some examples and problems are set apart from the rest of the text. These may be examples of problems that will be solved in the notes, or complete problems that you might try on your own:

> Problem: Write a program that will answer the ultimate question of life, the

universe, and everything.

Finally, some interesting or relevant facts which do not quite fit with the rest of the text are given as asides:

> You can learn a lot from these asides. Sometimes, they tell you about the requirements of the course, while other times, they point out exceptions to what you've just learned; on some occasions, they just tell a story.

Evaluation and grading

Please note that all of the graded work in this course (assignments, quizzes, midterm, and final exam) is to be done individually. Collaboration on that work is prohibited. Additional exercises, which are not assigned marks, will be made available for group participation.

Please note: All final grades are subject to departmental review.

Assignments

There will be six assignments named assignment 0 through 5. The first assignment, assignment 0, will not count for marks, but it will guide you through setting up a Java programming environment on your computer and submitting assignments through the web site. This makes it a prerequisite for the other assignments.

The remaining assignments will consist mostly of programming questions; there may also be questions that require brief written responses. The details for each assignment are found in the assignments section of the course.

A typical assignment will consist of one or two programming problems. A description of a problem in English will be presented; for example, create a Java program to input a list of product prices and output the average and maximum prices. You will then be required to write a complete Java program, compile it, and run it. There may also be questions that require short written answers; for example, briefly explain the error in a given program.

It may not be possible for assignments to be marked and returned before the next assignment is due. Be prepared for that possibility; the quizzes (described below) will offer more immediate feedback on your progress in the course.

All assignments must follow the assignment guidelines given elsewhere in this introduction.

Quizzes

In addition to the assignment, each unit will conclude with a brief quiz. These quizzes serve two purposes: to help you evaluate your progress in the course, and to prepare you for some of the kinds of questions that may appear on the midterm and final exam.

The quizzes will be brief, timed, and multiple choice. The quiz for a unit may be written any time in the week before that unit's assignment due date, up until the date and time when the assignment is due. The results of your quiz will be available after that date. It is not possible to write late quizzes or make up

missed ones.

Only the highest 7 out of 10 quiz marks will count towards your final mark.

Participation

These marks are given to encourage participation in the course. In the past, many students have taken an anonymous approach to the course and have not reached out for help when they needed to. For this reason, a total of 5% will be awarded based on your level of participation in the online class.

You will receive credit for reading all of the discussion group messages, posting an online introduction about yourself (details will be posted in the *Introductions* discussion group), for attempting all of the assignments and quizzes, and for posting other course related messages in the discussion groups. Basically, you will get course credit for asking (and even answering) questions about the course.

Students registered in the flexible study section of the course, will receive some portion of their participation grade simply for attending the tutorial sessions.

Midterm

There will be a single midterm test in the course, scheduled by your instructor. It will come in one of two forms, at the instructor's discretion. It may be a **timed 70 minute online** test that must be written within a prescribed set of hours on the scheduled day. Alternatively, it may be in the form of a **take-home test**, where the questions are released, you may work on the test on your own time, and submit it to the instructor by a particular date and time.

The test will consist of two parts. Part A will be short (written) answer and/or multiple choice questions. Part B will consist of programming problems.

The midterm will cover the units completed by the date of the midterm. The precise extent of the material covered will be described by your instructor prior to the test.

Final exam

The final exam will be written at the University of Manitoba (UM), Fort Garry campus or at an approved off-campus location. **Students needing to write at an off-campus location must declare a location by the specified deadline date** (see off-campus declaration and policy under Student Resources on course homepage). **Students writing at the UM Fort Garry campus do not need to declare an exam location.**

The Registrar's Office is responsible for the final exam schedule which is available approximately one month after the start of the course.

The exam will consist of two parts. Part A will be short (written) answer and/or multiple choice questions. Part B will consist of programming problems. Your instructor will provide you with more details about your exam.

The final exam will be cumulative. That is, it will cover all of the material in the course. There may be a greater emphasis on material learned after the midterm.

Distribution of marks

Item	Percentage
Assignments (1 through 5, each worth 3.6%)	18%
Quizzes (best 7 out of 10, each worth 1%)	7%
Participation	5%
Midterm	15%
Final Exam	55%
Total	100%

Assignment guidelines

All assignments submitted must follow these guidelines:

- **Important** A Student Honesty Declaration must be signed before your assignments will be graded. To view (and then print) the required form, click here. The form must be signed and submitted to the dropbox on or before the due date of the first assignment. According to regulations, your assignments will not be marked without the signed declaration. Instead, you will receive a grade of 0.
- **Late assignments will not normally be accepted.** Extensions will be granted only in exceptional circumstances. Contact your instructor about any conflicts well in advance of the due date.
- Information about changes related to assignments and the course will be posted to the course home page and the course discussion boards. You should check this site often (at least twice a week). It is **your responsibility** to become aware of any such changes by checking the site on a regular basis.
- Sample solutions to programming questions will be available on the web after the assignment due date. Students are expected to review the solutions on the web. Getting full marks on an assignment does not mean that the assignment is perfect; it is impossible for the marker to check each line of each assignment. The markers do the best they can in the few hours allocated to them. You are liable to lose marks for wrong or unacceptable programming on a test or exam even if you did not lose marks for such on an assignment.
- Penalties and disciplinary action for deficient term work: You must obtain at least 40% on your combined assignment and quiz marks (10 out of a possible 25 term marks); otherwise, your final letter grade will be reduced by one full letter *after* it has been calculated (e.g., an earned B+ becomes a C+).
- Even if your programs are not 100% complete or error-free, submit them. Partial marks will be awarded for reasonable attempts.

Assignment marking scheme

Element	Marks	Requirements
Standards	2	All assignments submitted must adhere to these programming standards.

Algorithm generally correct	4	Does the program answer the question? Could the program abnormally terminate because it was not written carefully enough? For example, are there checks for zero division, array out of bounds, etc.
Efficiency and Elegance	2	Marks may be lost for undesirable features such as unnecessary looping, excessive use of memory (e.g., using arrays when not warranted), and design flaws (e.g., duplicating code rather than using methods).
Output	2	This is basically a check to ensure that the program was run with the required data. If the output is wrong because of an incorrect algorithm, you will not be penalized twice. On the other hand, if no output is produced because of a deficiency of the program, then these two marks will be lost.
Total	10	

Academic dishonesty

Academic dishonesty is a very serious offense and will be dealt with in accordance with the University's discipline bylaw. Examples of academic dishonesty include:

- submitting assignments which are not entirely your own work;
- letting others copy work that you created;
- use of unauthorized material during a test or examination;
- writing an examination for another person; or
- having another person write an examination for you.

Please see section 8 of the General Academic Regulations and Requirements in the University of Manitoba general calendar for details.

Acknowledgments

Content specialist: John Braico, B.Sc.,MA
Computer Science
University of Manitoba

Instructional designer: Richard Jones, BES, AGDDE(T), MDE
Distance and Online Education
Extended Education
University of Manitoba

Editor: James B. Hartman, Ph.D.
Distance and Online Education
Extended Education
University of Manitoba

Desktop publisher: Lorna Allard
Distance and Online Education

Extended Education
University of Manitoba

Web Developer: Janice Miller, M.Sc.
Distance and Online Education
Extended Education
University of Manitoba