# APS 105 – Computer Fundamentals

Project Parts 1 and 2: Topic Selection and Proposal

Fall 1999

(To be completed *before* your lab period, week of November 8.) This proposal is worth 20% of the final project mark.

---

## 1 Overview

For the final programming assignment of the course, you are to choose and implement a program of your own choice. The development of this program, using good software engineering practices, will be used to fulfill the project requirement for the course. The development process that you are to use consists of the following tasks:

1. Topic selection done via email.
2. Generation of a project proposal describing the software you are going to develop and an outline of its requirements.
3. Generation of a formal specification document for your software.
4. Implementation (coding) of the specification.
5. Thorough testing and debugging of the implementation.
6. Generation of a summary document that describes how your implementation satisfies the specification.
7. Demonstration of the completed software during the final lab session.

Each student will work individually on the project, as no group projects are permitted. If you have any help with the project, or use any code (such as libraries or API components) which you did not write, you must clearly identify this unoriginal work in your submissions. It is an academic offense if you do not do so.

The project is worth 10% of your final course mark, with the marks for the project being allocated as indicated in the table below.

| | |
|---|---|
| Proposal | 20% |
| Specification | 30% |
| Demonstration | 30% |
| Implementation Report | 20% |



## 2 Topic Selection

There are no limitations in your choice of topic. However, since it is intended that the project will require three weeks to fully specify, design, implement, test and document, a good rule of thumb is that your program should be about three times as complex as one of the later labs. Some examples of an appropriate project are:

1. 2 player chess game (optionally with a graphical user interface)
2. A screen saver (This is more difficult than you might think)
3. A hotel reservation system

Try and select a topic that is within your abilities. It is better to successfully complete a simple project than it is to only partially complete a more ambitious one. Your program must be robust, and will need to provide some form of reliable user interface. You do not need to use the graphical interface provided by java's `AWT`, but you are certainly encouraged to do so. If you have difficulty finding a topic there will be a list of possible topic ideas maintained on the course web-page: `www.ecf.utoronto.ca/~aps105/`

Every student must select a *relatively unique* topic that has not already been selected by some other student. To enforce this restriction, you must obtain permission to write a proposal on the topic you have selected. To obtain permission you will have to send email to:

`czajkow@ecf.utoronto.ca`

Your email message **must** have as its subject line:

`APS105 PROJECT:` <*topic-name*>

where you substitute a simple one or two word description of your topic for <*topic-name*>. Students will be given permission to use topics on a first-come, first-served basis. Once you have obtained permission, you will then need to generate a written submission, based on the requirements described in the next section.

## 3 Requirements For Writing the Proposal

Once your choice of topic has been approved, you are to create a one page proposal describing your selected program. You will need to indicate the purpose of the program and explain how it is to be used. You should also provide some indication of the motivation for your particular topic – why is it interesting, worthwhile or a good learning experience?

The proposal will need to outline the program's requirements. That is, you should describe the nature of the input and output and the conditions that the program is intended to be used in. Describe the general operation of the final program, detailing what it will be able to do, as well as any of its limitations. You may find it useful to indicate any significant features regarding the design of the final program, such as the use of the `AWT`. You do not need to be too detailed as part 2 of the project is a complete, formal specification for the program.

Your proposal must not exceed one page in length, in a font no smaller than 11 point. Handwritten submissions will not be accepted. The submission may include any sketches, diagrams and pseudo code that you feel is appropriate, provided that you do not exceed the one page limit on length. You must also clearly display your full name and student number on the top of the actual proposal and not on a separate cover sheet. Ensure that you make at least two copies of your proposal, so that you have a copy available when doing the later parts of the project.

Do not assume that your proposal will be accepted and immediately begin specifying or implementing your project. The teaching assistant who evaluates the proposal may determine that your topic is unsuitable

for any one of a variety of reasons and will require that you modify the existing proposal or select an alternative topic. You must ensure that you have the same teaching assistant mark all four components of your project. This will assist them in gaining familiarity with your project and thus better enable them to help you with any difficulties you are encounter.

## 4 After the Proposal: Specification and beyond

You will receive additional instructions at a later date concerning how to write your specification and complete the rest of your project.



# Sample Proposal: The Game of Life

Anthony Cox (99123456)

November 8, 1999

For my project, I would like to implement the game of Life. I have always been interested in the game since I first saw it on a Windows based system several years ago and would like to have my own version which runs on any system. For this reason, I would like to program it in Java so that I gain this portability. The implementation of this proposal will improve my understanding of java since it extends the material of the course by using the abstract windowing toolkit (AWT), mouse input, file input, and thread based concurrency.

The game is quite simple and is based on an arbitrary sized window onto an unbounded grid. For the purposes of simulation, anything which takes place outside of the window is ignored. Each cell on the grid represents either a dead (white) or an alive (blue) organism, and has as its neighbors the set of 8 cells surrounding it. On each turn, the current grid configuration is used to make up the next grid configuration with the following rules applying:

1. Any living cell with $< 2$ or $> 3$ living neighbors dies in the next turn.
2. Any living cell with 2 or 3 living neighbors remains alive in the next turn.
3. Any dead cell with 3 neighbors becomes alive (is born) in the next turn.

The program will use the AWT to display the board, with a grid of 20 by 10 cells. Three buttons will be provided on the bottom of the window: one to start the game running, one to stop it, and one to display a 'help' screen. A timing thread will update the grid approximately once per second after the start button has been clicked. If all the cells die, the game will exit on its own, otherwise it will exit when the stop button is clicked. The help screen will describe the rules of the game, so that users understand the program.

The initial grid configuration will be read from a file whose name is a command line parameter for the program. No interactive way to set up the grid will be provided. If no file name is present, the default name `lifedata` will be used. Instructions on how to setup other input files will be provided in a file named `README`.