<u>ACADEMIC CONDUCT</u>

It is expected that all work handed in by a student will be original work that has been done by the individual. If it is not, then this act of intellectual dishonesty will be dealt with severely. An *Academic Dishonesty Incident Report* will be filed with the Office of Student Conduct. If a record of previous academic dishonesty is established, the case will be forwarded to the Academic Integrity Board. The complete process is described in Mount Royal's *Code of Student Conduct.*

Normally, the sanction for any student accused of cheating will be zero on the assignment (in the case of one student giving part of his/her assignment to another student, both students are considered to be cheating). Sanctions for further incidents of cheating by the same student will be reviewed by the Academic Integrity Board and may ultimately result in expulsion from the College.

While students are expected to work reasonably independently, we do not expect you to work in isolation. Often you learn best when working with others on an assignment. So what degree of collaboration is expected and, indeed, encouraged, and what is deemed to be cheating?

In general, we encourage things like bouncing ideas off one another, discussing which of two alternate solutions might be better (and why), and getting another's ideas on how to resolve a difficulty that you have already spent time on. You should not be working so closely together that someone else's solution becomes incorporated into your product. These general guidelines apply to **<u>any type</u>** of assignment. Some more detailed guidelines for programming assignments follow.

There are two main steps in creating a computer program—the first is to determine how to solve the problem (by understanding all the nuances of the problem and then designing a solution), the second is to write the program and thoroughly test it. You need to become proficient in both areas.

In the first step, you need to understand the problem. First, analyze the problem yourself and try to understand it as much as possible. Then, if you like, discuss your ideas with others—this way you can learn from each other. Once you feel you understand the problem, decide how to approach solving it. First come up with a high-level design yourself. After you have the first cut at a design, you may want to compare notes with others to see if you've missed anything, or to help you resolve a difficulty you have with your design. However, you **<u>MUST</u>** come up with your own design first; if not, you will never acquire the skills required of a good programmer.

When you get to the point of writing the actual program, you must work independently. It is **<u>NOT</u>** acceptable to be coding with others. Write your program by taking your design, possibly fleshing out some details, and writing the code in the appropriate programming language. If you have difficulty with a certain statement, check your notes or your text. If you are still having problems with the code, then you can ask others for help. However, you must master the language as quickly as possible, and not rely on others. When testing the program, if you are having problems with a section and have spent some time trying to find the problem yourself, it is a good idea to ask others if they can help you. Other people will often see errors that you cannot see because you are too close to your solution – that is, you no longer see it clearly. Don't forget that if you need help at any time, the Instructional Assistant and your Instructor are available.

When complete, two students' programs should be essentially independent of one another. Each student should have attempted each step of the problem solving method alone before discussing it with others. In this way you can develop your own skills, while still learning from (and helping) others.