GPR-200 Introduction to Modern Graphics Programming

Fall 2020

Sections:	**GPR-200-01**:	Mon & Thurs,	12:45pm – 2:00pm
	GPR-200-02:	Mon & Thurs,	2:20pm – 3:35pm
	GPR-200-03:	Tues & Fri,	12:45pm – 2:00pm
Classroom:	Google Meet (check your calendar)		

Course Instructor:

Daniel S. Buckstein

A graduate of UOIT in Ontario, Canada, Dan holds a Master's degree in Computer Science and a Bachelor's degree in Information Technology, majoring in Game Development & Entrepreneurship. He has a keen interest various areas of mathematics, including calculus and linear algebra, both of which carry over to computer animation and graphics. Dan also practices low-level programming and building clean, modular code and frameworks. This has been demonstrated in a long-standing passion for creating high- and low-level graphics and animation tools in both industry and academic contexts. His programming and development interests include game engines, tool design, virtual reality and augmented reality. Dan's passion for programming intertwines with his passion for teaching and he is always looking for innovative ways to deliver course content.

Contact Info: **dbuckstein@champlain.edu**
Office Hours: **Google Meet (check Canvas); Mon,Tues,Thurs,Fri, 10–11am**

Course Description:

This course takes a shaders-first dive into modern computer graphics programming. Students are introduced to vertex and fragment shaders, 2D and 3D coordinate spaces, drawing primitives, lighting and shading, data flow and manipulation, and modern GPU capabilities. Linear algebra and 3D math concepts will be refreshed and/or introduced for applicable topics.
Prerequisite courses: CSI 240 - Advanced Programming (C or better)
Credit Hours: 3

Course Design:

This course focuses on both theoretical and practical aspects and applications of modern computer graphics programming. The primary takeaways include a fundamental understanding of modern graphics concepts, and how they are applied. Theoretical content will be delivered to students through lectures, discussions and tutorials, taking place during the scheduled class time. The hands-on component requires students to program shaders and graphics algorithms and systems using OpenGL, GLSL, C/C++, or other engines where applicable. The scheduled class time will consist of a mixture of theoretical discussions, hands-on programming activities and code reviews. Students are expected to attend and participate in all scheduled classes. Furthermore, it is highly recommended that you practice shader programming daily to keep up with practices discussed in class. This will enforce your algorithm development skills and understanding of the course materials.

[0]Any part of this syllabus or related documents is subject to change, based on the instructor's discretion.

Course Objectives:
Upon completion of this course, students will be able to:

- ...describe modern rendering techniques
- ...program simple windowed applications using modern OpenGL
- ...implement industry-standard shaders using the OpenGL Shading Language (GLSL)
- ...utilize linear algebra, vectors and matrices in the graphics context
- ...explore 2D and 3D graphics and animation concepts

Evaluation Method:

The points for each deliverable (see category descriptions below) will be distributed appropriately between the following metrics, each weighted by importance and contribution to overall grade:

- **Implementation: Architecture & Design (20%)**: All evaluations require practical knowledge of architecture and design, within the context of the course content, using various programming languages and frameworks.
- **Implementation: Content & Material (20%)**: All evaluations require practical knowledge of the course subject matter, content, materials, etc. (e.g. implementing algorithms).
- **Demonstration: Presentation & Walkthrough (20%)**: All evaluations require presentation and/or interpretation of the materials, outputs, products, results and/or scenarios.
- **Demonstration: Products & Output (20%)**: All evaluations require evidence of having produced and/or interpreted a visibly correct and functional product.
- **Organization: Documentation & Management (20%)**: All evaluations require evidence and explanation of standard code organizational practices (e.g. commenting, version control) and how they help in the context of each evaluation.

	Deliverable Category	**Occurrences**	**Grade Weight**
1.	**Readings & Quizzes**	5% × 8	35%*
2.	**Lab Assignments & Demos**	5% × 8	35%*
3.	**Midterm Project & Demo**	10% × 1	10%
4.	**Final Project & Technical Interview**	20% × 1	20%

*The lowest score in these categories will be dropped.

Evaluation Method Descriptions:

1. Readings & Quizzes (35%*):
Quizzes may be assigned weekly throughout the term at the beginning of the first class each week. The quizzes will consist of multiple choice, true/false, short-answer and essay questions. Questions are pertinent to upcoming lessons and content and may include a brief review of past content. The intent of quizzes is to ensure that you are exposed to core terms and concepts, and attain the foundational knowledge needed to proceed with the following lab assignment (see below). **You must keep up with the class activities and readings to-date**. Quizzes will be administered online in Canvas and are **strictly independent evaluations**. Each quiz may also have a video recording component.

Implementation: Answer basic questions pertinent to shaders and graphics algorithms.
Demonstration: Analyze and interpret provided graphics scenarios.
Organization: Think critically and explain issues pertinent to graphics programming.

2. Lab Assignments & Demos (35%*):
Lab assignments are designed to provide quick yet effective opportunities to gain hands-on experience with the weekly subject matter. Due to the broad nature of the course, some of the lab assignments may be selected by students from a pool of topics designed by the instructor. All potential lab activities will build off of prior activities, therefore it is important to do them all and seek help where needed. Labs pertain to recent materials and require programming in C/C++ and other APIs relevant to the subject matter (e.g. GLSL, animation algorithms). The result of each assignment will be a working implementation of a fundamental algorithm or principle discussed in class. Activities may also help you familiarize yourself with the course framework's capabilities. Assignment descriptions and requirements will be provided on Canvas, with some instructions on what needs to be done, but not necessarily how to get there. Demonstrations will be submitted as pre-recorded videos (see below) and are designed to get you to think about what you have done and explain it (which will help you with your interviewing skills). Students may work **in pairs (recommended) or independently** to complete each assignment. Labs may be due as early as one week after being assigned.

Note: Do not jump straight into code! You should take a substantial amount of time to prepare for the lab (including note-taking, asking questions and drawing diagrams). You will find that this preparation time will help reduce the implementation time immensely by helping you predict issues before they occur. Per the organizational component of your grade, do whatever you can to demonstrate that you have put thought into your work before diving in. Complete a UML diagram to help bridge the instructions with your code design.

Note: Each lab should be considered a viable contribution to the term projects (i.e. by completing a lab, you are inherently completing your other projects, midterm and/or final)! The core point and takeaways of each lab (i.e. why it is relevant and pertinent in the course and field) will be provided in the briefing.

Implementation: Gain hands-on experience developing shaders and graphics algorithms.
Demonstration: Higher-level show and tell of finished product and walk-through of code.
Organization: Document all code produced (read-me, UML, etc.); maintain code base using version control.

3. Midterm Project & Demo (10%):
Projects will require students to implement one or more fundamental algorithms in areas previously discussed in the course; tasks will require more time and critical thought to complete than labs. Students may work **in pairs (recommended) or independently** to complete the project. The projects will be graded in the same fashion as the lab assignments. Project descriptions and requirements will be provided in separate briefing documents on Canvas. Specific instructions **will not** be provided, but a certain level of expectation will be indicated. These pieces are designed to make excellent portfolio contributions, so take each project seriously!

Implementation: Gain hands-on experience developing shaders and graphics algorithms; produce portfolio-worthy content.
Demonstration: Higher-level show and tell of finished product and walk-through of code; discuss product in greater detail.
Organization: Document all code produced (read-me, UML, etc.); maintain code base using version control.

4. Final Project & Technical Interview (20%):
The final project is selected from a pool of instructor-designed topics. Students may also propose their own project to the instructor before beginning work. The outcome of the project is a **portfolio contribution** completed **in pairs (recommended) or independently**. This project is designed to be flexible and creative, but students should also strive to use this opportunity to help complete Capstone or Production projects. Potential project proposals include, but are not limited to: investigation and implementation of an intermediate to advanced algorithm in the field; building a custom tool for an industry-grade engine that will facilitate the relevant pipelines (e.g. art-to-screen, data flow, etc.); building a stand-alone system that improves upon existing systems; and more! The final project will be evaluated through **demonstration** in the form of a **technical interview**; students are expected to show the product, discuss the code in detail and be professional; this is meant to emulate a job interview! The final project may be assigned throughout the term in stages including, but not limited to: initial proposal and the final interview.

Implementation: Gain hands-on experience developing shaders and graphics algorithms; produce portfolio-worthy content; design and develop major contribution that applies all learned material; demonstrate practical knowledge in an interview format.
Demonstration: Higher-level show and tell of finished product and walk-through of code; analyze and discuss more advanced scenarios and what-ifs pertinent to product; present live demo of code and product.
Organization: Document all code produced (read-me, UML, etc.); maintain code base using version control; produce relevant interview materials.

Evaluation Policies:

Important: Pass Clause

An average of ***50% in each deliverable category*** is required to receive a passing grade.

Version Control

Some form of **version control** must be used consistently and effectively for all lab and project evaluations. Popular version control software includes *Git* and *Mercurial* command line interfaces; each has a variety of supplementary GUI-based software available (e.g. *Tortoise Hg* for Mercurial, *SmartGit* for Git). There is also a variety of hosts for remote (online) repositories, such as Bitbucket (Git and Mercurial), GitHub (ubiquitous for Git) and Pineapple (Champlain-owned). Failure to use version control will result in a substantial grade penalty. Class and assignment starters may be delivered using an instructor-managed repository. Submit a link to your repository, either public or shared with the instructor's account, either once per assignment or whenever a new repository is started; branching can be used to organize multiple assignments worth of work in a single repository. The use of version control does not waive other assignment submission requirements.

Note: *Ignore filters* exist for a reason; do not track ignored files. If you are unsure of which files should and should not be tracked, *please ask.*

Note: *Branches* exist for a reason; do not complete everything on one branch. If you are unsure of how to use branches, *please ask.*

Note: *Deadlines* exist for a reason; do not wait until last minute to submit your repository info. If you have any questions about assignment submissions, *please ask.*

Frameworks & Resources

All in-class work and assignments will be completed using one or more types of frameworks (e.g. AAA, indie, web-based) using a variety of languages (e.g. C/C++/C#/GLSL). The framework may be determined in each assignment briefing. The purpose of any framework is so that you do not need to start from zero; you will have some form of graphics and other typical engine components at your disposal. Regardless of framework choice, data- and object-oriented programming practices must be observed; you may not use visual programming interfaces such as Blueprint. For example: if using Unity, front-end work is done in C# with the option to write a C/C++ plugin back-end. Another example: you may be provided with a custom framework written in pure C.

Grading

Grading for all lab assignments will happen through **demonstration** in some form, which is a good opportunity to think about what you have implemented and explain your thought process directly. Demos may be in-person in class or pre-recorded; please consult each assignment briefing for specific submission instructions. Regardless, a demo should take no more than **five minutes** and include a brief show-and-tell of the assignment. Be prepared to show all pertinent components of the assignment (anything specified as to-do in the briefing) and explain how they function. All assignments and projects must also have a Canvas submission, which may or may not be used for further evaluation at the instructor's discretion. Failure to submit the required info for an assignment in Canvas will result in a grade of ***zero***. Late submissions will not be accepted and will result in a grade of ***zero***. There will also be substantial penalties for inadequate use of version control, lack of documentation, violating coding standards specified in the assignment briefing, use of outdated techniques, etc. Specific grading requirements will be provided in each briefing.

Documentation

All contributions to an evaluation should be documented by the author(s). For example, any algorithm implemented can have a block comment above it that describes the step-by-step breakdown of how it works (in English or pseudocode). Documentation also includes the use of descriptive and meaningful comments in version control commits. Each project should also include a read-me and a UML diagram of new systems. The lack of adequate documentation in code may result in grade reductions and a longer wait time to get your grade back!

Deadlines

Deadlines for all evaluations may change with the discretion of the instructor. To avoid missing deadlines, begin work as soon as it is assigned and submit your work at least 24 hours prior to the deadline to avoid rushing to submit and to leave time to fix errors. Canvas submissions will close at the same time as the respective deadlines. Late submissions will receive a grade of **zero** and **email submissions will not be accepted**. One more time: **email submissions will not be accepted**. Extensions and make-ups will not be granted unless deemed necessary by documented extenuating circumstances (e.g. a medical emergency).

Asking for Help & Attending Office Hours

The office hours listed above are times only the *bare minimum* times to drop in and ask for help; they do not in any way imply that your instructor is not available outside these times. If you need help outside of the hours listed above, please send an email requesting an appointment. Office hours and appointments are useful for clarifying details about class materials and assignments, getting reliable information, interpreting the syllabus, and chatting about future courses and subject matter. Also note that office hours are for clarifying problems in class, not getting a repeat lesson because you missed class. Please make the best of your education!

From the assignments alone, you are expected to figure out the 'how' in "how do I make this work" and this will not openly be disclosed by asking. You will draw these conclusions from the 'what' presented in the assignment briefing, the 'why' presented in class, deep discussion of 'when' certain techniques are appropriate, and 'where' to find information, guiding your independent research. The 'who' is you, a programmer!

If you attend office hours to ask for help with an assignment, be prepared to demonstrate and explain your process thus far and provide a direct description of the problem. If you are working in a team for an assignment, all members of the team must be present to ask for help; this demonstrates that you have made an effort to communicate internally.

Debugging requests of any sort via email will be ignored; this is not an appropriate way to solve a problem and is also extremely difficult since we are dealing with living code. If you have a persistent issue that you need help with, make an effort to attend office hours. Furthermore, when visiting office hours to debug, bring your own laptop and/or ensure your repository is up-to-date.

Finally, asking for critical help with the fundamentals of an assignment within 48 hours of its deadline is last-minute and is simply unacceptable. Last-minute help will only be considered if conversation about the assignment has been sustained since the assign date. Always begin working on an assignment the same day it is assigned so you may identify potential problems immediately.

Note: Failure to appear at a scheduled appointment, unless due to a documented emergency, will result in a grade penalty of 1% per instance. If you need to cancel or reschedule, please do so in advance by sending an email.

Zero Tolerance for Plagiarism

Plagiarism is the act of taking credit for graded work that is not yours. Copying a completed algorithm off of a website or out of a book and calling it your own is also plagiarism. Copying your own prior work for evaluation also counts as plagiarism. You may always ask others for help, but asking to view another student's work is a slippery slope towards plagiarism; do not do this, instead, discuss the problem with your peers and see if you can agree on a solution. It is entirely up to the party with the problem to solve the problem.

To avoid the risk of plagiarism in your work, ensure that all contributors label their own work within a submission (see 'Documentation' above). In addition to this, ensure that all borrowed material (e.g. from the internet or a literary resource) is credited appropriately. This course does not mandate the use of a standard citation format (such as APA or MLA), so it will be deemed acceptable to add a URL and/or article information (author, title, year, etc.) directly inline with code where the reference is made. **Any portion of code cited from another source will not count toward your grade**, since it is not original.

There are many reasons why you should not plagiarize. Reasons include, but are not limited to:

1. You are not learning anything by copying others' work with little to no understanding of what you are copying and why.
2. Plagiarism reduces the value and integrity of your degree for you and for your classmates.
3. Your classmates are your future colleagues and industry connections. Do not leave them with a negative impression of your work ethics; ***they will remember you***.

This course has a straightforward and ***strict policy of zero-tolerance for plagiarism***:

1. All parties involved in a case of duplicate or highly-resemblant submissions will be called in for a meeting to discuss the work in question.
2. If you are caught and confirmed plagiarizing in any form, following the usual process your **final grade in the course** will be immediately recorded as "**F**". You may visit the registrar's office to replace this with "W" if the incident occurs before the withdrawal deadline.

Please see Champlain College's Academic Honesty Policy summary below, and the complete policy posted on Canvas. Do not plagiarize. Thank you for your understanding and cooperation.

Avoiding Headaches for Everyone

The easiest way to avoid headaches for you and your instructor is simple: ***read and follow all instructions***. For example, if a project requires you to submit certain files, ensure that these files are submitted. Similarly, if a project asks you to omit certain files, do not submit these files.

In addition to following all instructions, you must ***provide instructions with all submissions on how to use your product and how to navigate the code***. This is especially true if you choose to use a framework other than one provided to complete an assignment. Please pull your weight to minimize grading time; the more you do to help your instructor, the sooner you will get your grades back!

Finally, you must ***test your work before submitting***. Any code submission that does not build and run as expected ***immediately out of the box*** will receive a score of ***zero***. Code grading is painful; not following instructions, providing information about your submission and testing your product all result in delays and therefore longer grading times.

Attendance

You are expected to attend all of the scheduled meeting times for your section since we will cover fundamental topics each week. Furthermore you are expected to show up to class on time. If you are unable to attend a class or anticipate being late, please inform your instructor before the fact. Attendance will be tracked using Canvas's roll-call feature. Unexcused absences will result in a **grade reduction of 2%** for each instance. Unexcused lateness will result in a **grade reduction of 1%** for each instance.

Missing class and showing up late for class will inherently have a negative impact on your grade (e.g., no make-up quizzes, in-class activities). Furthermore it is unprofessional and will not be tolerated in the working world. Students are responsible for all missed work, regardless of the reason for absence. It is also the absentee's responsibility to obtain all missing notes or materials. Attendance refers to both physical and mental presence, i.e. listening in class and staying focused. If at a later date you ask questions about things covered in class because you've never heard them because you were busy playing games, you're on your own.

General Course Policies and Expectations:

- Put away your phones and other mobile devices for the duration of class, in-person or online. They are are incredibly distracting to you, your peers and the instructor. If you are expecting a call, please excuse yourself. Distracting use of such devices will result in you being asked to leave class. Your attendance is a privilege, not a right; don't waste it.
- Using computing devices for non-class related activities will result in scathing penalties. For your sake and those around you, don't do it. This also applies to online learning.
- Install all required software and packages provided by the instructor. Failure to do so will greatly hinder your progress in the course.
- **Do not** package up your project's working directory with a bug (e.g. "because it just won't work") and expect your instructor to fix it via email. Email your instructor with **specific questions and evidence that you have attempted to solve the problem**. Attend office hours as needed.
- Independent Assignments: Students are expected to work independently. **Viewing**, **offering** and **accepting** materials are acts of **plagiarism**, which is a serious offense and **all involved parties will be penalized according to the course's and the College's academic honesty policy**. Discussion amongst students is encouraged, but when in doubt, direct questions to the professor.
- Group Assignments: Students are expected to work collaboratively. All students are expected to contribute equally and are to respect the views of other students. The professor reserves the right to distribute grades according to contribution. Please pull your own weight in your group projects and do not let others suffer due to a lack of organization or interest.
- Code grading may take a few weeks, depending on the submission load across all of your instructor's sections. Code grades returned in a few days is unlikely and should not change this expectation. All questions regarding any graded work **must** be brought to the attention of the professor within **one week** of when the grade is returned.
- I expect you to ***read the syllabus and all related documents***. Asking for clarification is acceptable, but asking questions that are explicitly answered in any of these documents will result in a reference to said documents (e.g. Q: "When are your office hours?" A: "Syllabus.").
- I expect you to seek out and find information that you might require as we do not have enough time to cover everything. Just reading the notes is not enough to get you an A. If you are thoroughly confused about any topic or concept, **go to office hours as early as possible**.
- I expect you to be programming ***every day*** or ***every two days*** at minimum in this course. Bring that passion and new discoveries with you to class. Ask questions when appropriate. Be curious!
- I expect that you have functional knowledge of essential mathematics for games and object-oriented programming concepts (classes, inheritance, virtual functions, etc.) as taught in previous courses; thus you should be reviewing these concepts.

Letter Grade Distribution:

			67.00 – 69.99	D+	
≥ 93.00	A	Work is stellar, exemplary.	63.00 – 66.99	D	Work is inadequate.
90.00 – 92.99	A–		60.00 – 62.99	D–	
87.00 – 89.99	B+				
83.00 – 86.99	B	Work exceeds expectations.	< 60.00	F	Work is incomplete
80.00 – 82.99	B–				or impertinent.
77.00 – 79.99	C+				
73.00 – 76.99	C	Work meets expectations.			
70.00 – 72.99	C–				

Tentative Course Outline:

All assignments are posted and labeled by a high-level overview of the subject matter covered. Lessons and tutorials will cover information pertinent to completing the assignment. See the 'Syllabus' page in Canvas for a dated list of tentative assignment due-dates and topics, the 'Assignments' and 'Quizzes' pages for a list of evaluations, and the 'Modules'page for a list of weekly materials (each week published in advance). All are subject to change depending on the timing of the course.

Data for Research Disclosure:

Any and all results of in-class and out-of-class assignments and examinations are data sources for research and may be used in published research. All such use will always be anonymous.

Syllabus Acknowledgment:

After reading the entirety of the syllabus and all supporting materials, please complete the "Syllabus Acknowledgment" quiz in Canvas. By acknowledging the syllabus, you are indicating that you have had the opportunity to address and resolve any questions or concerns regarding the syllabus content, and that you have read and understand all of the course requirements and policies described therein. **You must complete this acknowledgment to receive a grade in the course! Failure to do so within the first week of classes may inhibit your ability to access and complete course content!**

Academic Honesty Policy Summary:

Introduction
In addition to skills and knowledge, Champlain College aims to teach students appropriate Ethical and Professional Standards of Conduct. The Academic Honesty Policy exists to inform students and Faculty of their obligations in upholding the highest standards of professional and ethical integrity. All student work is subject to the Academic Honesty Policy. Professional and Academic practice provides guidance about how to properly cite, reference, and attribute the intellectual property of others. Any attempt to deceive a faculty member or to help another student to do so will be considered a violation of this standard.

Instructor's Intended Purpose
The student's work must match the instructor's intended purpose for an assignment. While the instructor will establish the intent of an assignment, each student must clarify outstanding questions of that intent for a given assignment.

Unauthorized/Excessive Assistance
The student may not give or get any unauthorized or excessive assistance in the preparation of any work.

Authorship
The student must clearly establish authorship of a work. Referenced work must be clearly documented, cited, and attributed, regardless of media or distribution. Even in the case of work licensed as public domain or Copyleft, (See: http://creativecommons.org/) the student must provide attribution of that work in order to uphold the standards of intent and authorship.

Declaration
Online submission of, or placing one's name on an exam, assignment, or any course document is a statement of academic honor that the student has not received or given inappropriate assistance in completing it and that the student has complied with the Academic Honesty Policy in that work.

Consequences
An instructor may impose a sanction on the student that varies depending upon the instructor's evaluation of the nature and gravity of the offense. Possible sanctions include but are not limited to, the following: (1) Require the student to redo the assignment; (2) Require the student to complete another assignment; (3) Assign a grade of zero to the assignment; (4) Assign a final grade of "F" for the course. A student may appeal these decisions according to the Academic Grievance Procedure. (See the relevant section in the Student Handbook.) Multiple violations of this policy will result in a referral to the Conduct Review Board for possible additional sanctions.
The full text of the Academic Honesty Policy is in the *Student Handbook*.

Diversity, Equity & Inclusion:

As a learning community, we should be working together to create an effective and respectful space; an environment that is safe and accepting, where everyone feels like they can bring their whole self to the table; a classroom that supports the ever changing and diverse learning needs and complexities of Champlain College students; a space that maximizes learning for everyone. Differences of opinion are encouraged and should be nurtured. Discussions themselves are collaborative learning ventures, and it is often through dialogue that we come to understand our own thoughts and positions.

We all co-create this positive, inclusive learning environment. The following are guiding principles for how this can be achieved:

- Appreciate that everyone is in a different "space" of learning: how they learn, different skill level, and the unique personal context they bring to the classroom. Each of us in this class will know more in some areas and less in others.
- Try not to make assumptions; ask questions to learn more about other perspectives, especially those that are different from your own.
- Lead with "generosity" when you listen to others' perspectives and prioritize kindness over proving the other wrong. Listen to learn first, rather than to immediately judge or "win."
- Respect others' perspectives and recognize your own biases and the limits of your own knowledge.
- Show respect for others as individuals by learning and using their preferred names and pronouns.
- Share your knowledge and skills with tact.
- Applaud each other's efforts and offer constructive feedback when possible.
- Respect the speaker, even when you do not agree with or support the point the speaker is making.
- Do not monopolize the conversation, rather, give others a chance to contribute.
- Realize that no opinion or position is out of bounds (bar hate speech), but all opinions should be articulated with thoughtfulness and humility.
- Take into consideration the impact of your words on others.
- Racial, ethnic, gender or other identity-based slurs are nothing other than hurtful and will not be tolerated.

Students with Disabilities:

If you believe that you have a disability requiring accommodations in this class, please contact the Coordinator of Services for Students with Disabilities as soon as possible. You will be able to schedule a meeting with an accommodations officer and have your documentation reviewed. During that meeting Skip or Denise will provide you with letters for your faculty, which will detail your needed accommodations. It is the student's responsibility to seek and secure accommodations prior to the start of an exam or project.

Contact: Office of Accessibility
Office: Skiff Hall 112
Phone: 802-865-5764
Email: accessibility@champlain.edu

Class during an Extended Campus Closure:

Champlain College is taking precautionary measures to ensure that this class can continue in a “virtual environment” even during an extended emergency such as severe weather, contagious disease, physical infrastructure failure, campus closure, or similar incident. This course will continue either online through a college-provided learning management system (Canvas), or through some other process, unless cancelled. In the event of such an emergency, students are expected to continue instructor-designated class activities, as directed by the instructor. Due to the nature of the “virtual environment” learning activities may differ slightly from the on-campus course. In order for this emergency preparedness plan to be effective, you are asked to do the following:

Immediately

- Ensure that you will have a computer and Internet access at the location (home or other) in which you will reside during an extended campus closure.
- Prepare yourself with the basic skills of logging into Canvas via the my.champlain.edu dashboard, finding your course(s), and entering them.
- Participate in a “warm-up” online activity in the “virtual environment” when directed to do so by your instructor.

During an Emergency

- Test your Internet access immediately upon arriving at your chosen residence during the campus closure.
- Log into Canvas and enter your courses.
- Check for emergency information on Champlain College main website (www.champlain.edu) which will indicate the semester week and day on which college classes will resume online.
- Enter your class and go to the appropriate week of class where you will receive directions from your instructor.