AP 2/9/10 09-47
Info 2/23/10

Undergraduate Distance Education Review Form

(Required for all courses taught by distance education for more than one-third of teaching contact hours.)

Existing and Special Topics Course

Course: COSC 110 Problem Solving and Structured Programming

Instructor(s) of Record: Waleed Farag

Phone: (724)357-7995 **Email:** farag@iup.edu

Step One: Proposer

A. Provide a brief narrative rationale for each of the items, A1- A5.

1. How is/are the instructor(s) qualified in the distance education delivery method as well as the discipline?

   I am a Tenured Associate Professor of Computer Science who has more than fifteen years of teaching experience. I have been teaching Introductory Computer Programming Courses at IUP and other institutions of Higher Education for over a decade and have developed pretty effective teaching techniques for such courses. Regarding qualifications in teaching on-line, I have already taught and am scheduled to teach various on-line sections that our Department has offered. In all of my courses, the use of the Learning Management System (LMS) is a significant and integrated part of every course. I have also attended numerous workshops on LMSs and on-line pedagogy in which some of these workshops were multiday ones. Moreover, as a researcher I had played instrumental role in designing and administering a pioneer synchronous distance education system. I believe that such deep understanding of the underlying delivery technology and the long-term expertise in teaching programming courses make me a very suitable person to teach such an on-line course.

2. How will each objective in the course be met using distance education technologies?

   The online course proposal, attached, presents a variety of tools and techniques to be used to ensure that the course learning outcomes will be met using the e-learning technology. Below is a brief explanation of how each course objective will be met:

   a) **Use an integrated programming environment.** The course employs a widely used Integrated Development Environment (IDE) for developing C++ programs. The repeated use of this IDE will ensure students' understanding of various capabilities of such a system. Detailed handouts of how this software can be used will be posted as part of Laboratory Exercises. Other synchronous activities can be used to reinforce students' understanding of the IDE. Moreover, students can obtain such software on DVDs at no cost from our department through its Academic Alliance Agreement with Microsoft. Another option is to download a freeware version of this program called "Microsoft Visual C++ 2008 Express Edition" from Microsoft Web Site. This freeware fully suffices the purposes of this course. The attainment of this objective will be assessed through students' performance in projects and discussion activities.

   b) **Develop algorithms from user problem statements.** Several reading assignments will guide the students on how to approach a given specifications and introduce a proper algorithmic solution suitable for implementation in a programming language. These will mainly address the analysis, algorithm development and testing which constitute the first step in Program Development Life Cycle (PDLC). Various problem solving approaches will also be covered. These concepts will be further reinforced and assessed through Laboratory Exercises, Discussion Forum Topics, Programming Projects, Quizzes and Exams. On planned and regular basis, the instructor will post questions on the discussion forum dealing with algorithm design issues and all students will be asked to respond to these questions and to comment on other students' posting. The instructor will ensure that the discussion will be steered in the proper

direction and this way a community of learners will be developed. Such a community provides students with excellent opportunity to learn from each other and to acquire the expected knowledge and skills.

c) **Express the solutions to computer oriented problems in flowcharts and/or pseudocode.** A number of illustrative examples will be posted to explain in details how students can represent their developed algorithms as pseudocode or using a pictorial method such as flowcharts. Laboratory Exercises, Discussion Forum and other synchronous communication techniques proposed in the syllabus will also be used to ensure that students meet this objective.

d) **Give commands to compile, link, and run their own programs, including using common options.** These skills are now part of the use of an IDE and are mastered if students can use the IDE effectively. Therefore, the techniques discussed in **A2.a** will be used to ensure the acquisition of these abilities.

e) **Proficiently transform designs of problem solutions into C++ programming language.** Several activities will be employed to instill this essential skill in the students these include the exposure to large number of Code Examples, Laboratory Exercises, Programming Projects, Discussion Forum, etc. These activities will focus on the Implementation Phase of the Program Development Life Cycle (PDLC) in which coding and testing are the main concepts. The ability to transform algorithms to code will be assessed via programming projects and tests. Students will be given a problem specification and then be asked to deliver a properly working C++ programs that solve the given problem. Moreover, such a skill will be stressed through various discussion forum topics, chat sessions and other synchronous activities.

f) **Apply debugging and testing techniques to locate errors and determine the effectiveness of a program.** This skill will be the focus of many activities throughout the semester. Techniques range from simple ones such as the use of echo outputting to more powerful ones such as the use of the Debugging Tools in the adopted IDE will be constantly utilized in Laboratory Exercises and Programming Projects. Besides, synchronous activities such as the use of Chat Rooms and Live Classroom (LC) sessions can be beneficial in accomplishing this course objective. Student performance in projects and exams will be an evidence of the degree of accomplishment of this objective.

g) **Recognize and use the correct C++ programming language syntax.** A powerful and effective tool that the proposed syllabus uses to facilitate the achievement of this objective is called CodeLab, a Web-based tool for learning programming. This tool is an excellent environment for teaching students how to develop programs using the C++ language. It helps students to get very familiar with the syntax of various language constructs in addition to instilling into them various essential programming skills. ***CodeLab provides students with instantaneous intelligent feedback to all their activities, a crucial feature that will compensate for the absence of physical contact with the instructor.*** Successful completion of all CodeLab exercises is part of the course grade. The instructor will monitor students' progress on these activities and grades will be assigned in proportion to how much of these exercises were fully performed.

h) **Apply structured programming techniques including design approaches, mnemonic naming, use of documentation, and avoidance of excessive branching.** Through several reading assignment, labs, projects and extended discussions of these techniques, students will be able to master the use of structured programming in their written code. Such skills will be evaluated via projects and exams. Chat and LC sessions can also help with this objective in particular if a student has a concern or question that requires direct interaction with the instructor.

i) **Use these programming elements: variable declaration, use of data types and simple data structures (arrays and records), decision structures, loop structures, input and output for terminals and files, output form, and subordinate functions.** This objective covers a comprehensive list of programming elements and constructs and to ensure students will be able to employ these effectively all the activities proposed in the syllabus will be used. Moreover, the use of CodeLab and Discussion Forum topics designed to address each of these constructs will be very beneficial in accomplishing this objective. Student attainment of this objective will be evaluated through performance in projects, discussion activities and exams.

3. How will instructor-student and student-student, if applicable, interaction take place?

The main vehicle for both instructor-student and student-student interactions will be the Discussion Forum Activities. These activities are discussed in details in the proposed syllabus and briefly in Item A2 above. Several discussion forum topics are designed to carefully ensure maximum students' involvement and participation in these activities. That participation will be part of the course grade as described below. With the objective of fostering a community of motivated learners, the instructor will moderate these discussions, post leading questions, provide constant feedback on student postings, and ensure the discussion is steered in the proper direction. Other proposed tools for instructor-student interaction include email, phone calls during the online office hours, and chat and LC sessions. Additionally, email and chat and LC sessions can be used to enhance student-student interaction.

4. How will student achievement be evaluated?

Several evaluation criteria are presented in the proposed syllabus. These are:

- **Required programming projects will be submitted by students for grading.**
- **Exams will be used to evaluate students' overall understanding of various course concepts.**
- **The quality of student's participation in different discussion forum activities also represents an important method of assessing the degree of understanding of the course material and the levels at which course objectives have been accomplished. Some of the factors considered in grading various postings include correctness, style, depth, and comprehensiveness.**
- **A Web-based programming environment called CodeLab is another method. CodeLab provides the student with a unique environment for learning programming at their own pace. As an assessment tool, grades will be given based on completion of all assigned exercises which fairly accounts for differences in students' ability.**

5. How will academic honesty for tests and assignments be addressed?

All course materials will be available on Moodle. Each student has a secured individual account through which s/he can access course contents. Although, the mere use of LMS authentication will not guarantee academic honesty, it is an important method of enforcing access control. Several additional techniques will be implemented to address the issue of academic integrity those include the following:

- **For Assignments:**
  - **All student submissions will be scrutinized for duplicate or similar programming solutions.**
  - **If a research paper is required, an anti-plagiarism tool such as Turn-It-In will be used.**
- **For Exams:**
  - **Limiting the availability period of all online quizzes/exams.**
  - **Exam's duration will be set so that students only have the minimum time necessary for solving the exam. From the instructor's experience, this is a very effective technique.**
  - **Various security measures provided by Moodle will be used for instance, exams will be displayed in secure windows in which printing and other functionalities are disabled.**
  - **The selection of exam questions from large pool of possible ones along with randomization can present each student with a unique test which can circumvent almost all cheating attempts.**
  - **One other promising option is the use of Remote Proctoring technology. Through a funded grant, the Proposal writer has been experimenting with one of these systems called Secure Remote Proctor (SRP) and has made a number of presentations on that subject to ACPAC and others. This system captures students' video and biometric profiles and provides effective deterrence and enforcement mechanisms against cheating. If IUP decides to adopt such technology, it can provide a much higher level of academic integrity.**

B. Submit to the department or its curriculum committee the responses to items A1-A5, the current official syllabus of record, along with the instructor developed online version of the syllabus, and the sample lesson. This lesson should clearly demonstrate how the distance education instructional format adequately assists

students to meet a course objective(s) using online or distance technology. It should relate to one concrete topic area indicated on the syllabus.

Step Two: Departmental/Dean Approval

Recommendation: ☑ Positive (The objectives of this course can be met via distance education)

☐ Negative

Waleed Faraj Waleed Faraj 12/18/2009
Signature of Department Designee Date

Endorsed: 1/13/10
Signature of College Dean Date

Forward form and supporting materials to Liberal Studies Office for consideration by the University-wide Undergraduate Curriculum Committee. Dual-level courses also require review by the University-wide Graduate Committee for graduate-level section.

Step Three: University-wide Undergraduate Curriculum Committee Approval

Recommendation: ☒ Positive (The objectives of this course can be met via distance education)

☐ Negative

2/15/10
Signature of Committee Co-Chair Date

Forward form and supporting materials to the Provost within 30 calendar days after received by committee.

Step Four: Provost Approval

☒ Approved as distance education course ☐ Rejected as distance education course

2/23/10
Signature of Provost Date

Forward form and supporting materials to Associate Provost.

COSC 110 Syllabus of Record

I. Catalog Description

COSC 110 Problem Solving and Structured Programming 3c-0l-3sh

(For science, mathematics, and computer science majors, and for others who have a sufficiently quantitative orientation.) Basic structure of modern digital computers; problem analysis and computer solution using flowcharting and the C++ language. Exemption or credit by examination possible.

II. Course Objectives

Upon successful completion of this course, the student will be able to

A. Use an integrated programming environment.

B. Develop algorithms from user problem statements.

C. Express the solutions to computer oriented problems in flowcharts and/or pseudocode.

D. Give commands to compile, link, and run their own programs, including using common options.

E. Proficiently transform designs of problem solutions into C++ programming language.

F. Apply debugging and testing techniques to locate errors and determine the effectiveness of a program.

G. Recognize and use the correct C++ programming language syntax.

H. Apply structured programming techniques including design approaches, mnemonic naming, use of documentation, and avoidance of excessive branching.

I. Use these programming elements: variable declaration, use of data types and simple data structures (arrays and records), decision structures, loop structures, input and output for terminals and files, output form, and subordinate functions.

III. Course Outline

A. Introduction 2 hrs

1. History of computers
2. Components of a computer
3. Programming languages

4. Compilation vs. interpretation

B. The Programming Environment 4 hrs

1. Editing
2. Compiling
3. Linking
4. Options
5. Debugging
6. Redirection of input and output

C. Algorithm development using flowcharts/psuedocode 4 hrs

1. Software engineering method
2. Classic problems - maximum, minimum, sum, average

D. Basic input and output 1 hr

E. Data types 4 hrs

1. Constants
2. Variables
3. Expressions
4. Library files and functions

F. Simple Data Structures 7 hrs

1. One dimensional arrays
2. Strings as arrays
3. Multi-dimensional arrays
4. Records
5. Classic problems - searching, sorting

G. Use of decision structures 3 hrs

1. Single alternative
2. Double alternative
3. Multiple alternative
4. Nested structures

H. Loops 3 hrs

1. While loop
2. Do-while loop
3. For loop
4. Counting loop

5. Priming read loop

I. Programming language form 2 hrs

1. Syntax
2. Structured code
3. Documentation
4. Case sensitivity

J. Advanced formatted I/O including file access 4 hrs

1. Formatted input and output
2. File I/O
3. Sequential file processing

K. Testing and Debugging Techniques 1 hr

L. Functions 3 hrs

1. Argument passing
2. Returning results
3. Recursion
4. Testing a program system

M. Introduction to the Object Oriented Approach 1 hr

IV. Evaluation Methods

The final grade for the course is determined as follows:

50-60% Examinations. Three mid-term exams and the final - each consisting primarily of multiple choice, true-false, and short answer questions.

30-35% Programming assignments. There are approximately six programming assignments worth varying numbers of points that collectively count this portion of grade. Suggested tasks for the assignments include: linear search, sorting, 2-D array processing, interactive programming, simulation, sequential file processing, modularization.

10-15% Class participation and quizzes. This may be based on written questions, verbal discussions, computer lab sessions, or other form of interaction.

Suggested Grading Scale: 90-100% A, 80-89% B, 70-79% C, 60-69% D, 0-59% F

V. Required textbook, supplemental books and readings

Adams, Joel, Leestma, Sanford, and Nyhoff, Larry. C++: An Introduction to Computing. Prentice Hall. 1995.

VI. Special resource requirements

None.

VII. Bibliography

Staugaard, Andrew C., Jr. Structuring Techniques: An Introduction Using Turbo C++.

Perry, Greg. Turbo C++ Programming 101. Sams Publishing. 1993.

Friedman, Frank, and Koffman, Elliot. Problem Solving, Abstraction, and Design Using C++. Addison-Wesley. 1994.

Schildt, Herbert. C++ from the group up. McGraw-Hill. 1994.

Savitch, Walter. Problem Solving with C++. Addison-Wesley, 1996.

COSC 110: Problem Solving and Structured Programming
Proposed On-line Syllabus of Record

Proposer: Dr. Waleed Farag
Office: 338 Stright Hall
Phone: (724)357-7995
E-mail: farag@iup.edu
Web Page: www.people.iup.edu/farag
Online Office Hours: Tuesdays/Thursdays from 9:30 AM to 10:45 AM

Technical Help Info

IUP IT Support Center phone: (724)357-4000, Email it-support-center@iup.edu, and Web Site http://www.iup.edu/itsupportcenter/

I. Course Description

Credits: 3.00 Lecture: 3.00
Prerequisites: N/A
For Science, Mathematics, and Computer Science majors and for others who have a sufficiently quantitative orientation. Basic structure of modern digital computers; problem analysis and computer solution using flowcharting and the C++ language are covered. Exemption or credit by examination possible.

II. Course Objectives

Upon successful completion of this course, the student will be able to:

1. Use an integrated programming environment.
2. Develop algorithms from user problem statements.
3. Express the solutions to computer oriented problems in flowcharts and/or pseudocode.
4. Give commands to compile, link, and run their own programs, including using common options.
5. Proficiently transform designs of problem solutions into C++ programming language.
6. Apply debugging and testing techniques to locate errors and determine the effectiveness of a program.
7. Recognize and use the correct C++ programming language syntax.
8. Apply structured programming techniques including design approaches, mnemonic naming, use of documentation, and avoidance of excessive branching.
9. Use these programming elements: variable declaration, use of data types and simple data structures (arrays and records), decision structures, loop structures, input and output for terminals and files, output form, and subordinate functions.

III. Required Textbook/On-line subscription

- **Nell Dale and Chip Weems, "Programming and Problem Solving with C++ – Brief Edition", *Jones and Bartlett, 5th edition 2010. ISBN "9780763771515".***
- **This course also requires ALL students to subscribe to CodeLab**, a Web-based online service for learning and practicing computer programming. Because of its on-line nature, CodeLab can be accessed anywhere or at anytime as long as you have access to the Internet. A semester-long subscription costs around $20. Other systems can also be used in place of CodeLab. CodeLab can be accessed using the following link http://codelab1.turingscraft.com/codelab/jsp/login1.jsp.

IV. Moodle (Web Based Tools)

This course uses Moodle as part of its classroom environment. Moodle is an example of a Learning Management System (LMS). If you are new to Moodle, make sure that you can access your account and can efficiently use various tools and facilities available on that LMS. For technical help, please contact the IT Support Center (whose contact information is in the first page). All materials, notes, exams, etc. related to this course are posted on Moodle. Also, all assignments will be available on Moodle and all student submissions are to be uploaded into Moodle. Moodle can be accessed using the following link http://moodle.iup.edu.

V. IUP Electronic Mail

In addition to Moodle, the university's I-mail client at (http://imail.iup.edu) can be used as another method of communicating although I do strongly recommend that ALL your questions and concerns be posted on the Moodle Discussion Forum. **Please note that contacts from non-IUP email accounts will NOT be considered.**

VI. Course Outline

A. Introduction **2 hours**

- History of computers
- Components of a computer
- Programming languages
- Compilation vs. interpretation

B. The Programming Environment **4 hours**

- Editing
- Compiling
- Linking
- Options
- Debugging
- Redirection of input and output

C. Algorithm development using flowcharts/psuedocode **4 hours**

- Software engineering method
- Classic problems - maximum, minimum, sum, average

D. Basic input and output **1 hour**

E. Data types **4 hours**

- Constants
- Variables
- Expressions
- Library files and functions

F. Simple Data structures **7 hours**

- One dimensional arrays
- Strings as arrays
- Multi-dimensional arrays
- Records
- Classic problems - searching, sorting

G. Use of decision structures **3 hours**

- Single alternative
- Double alternative
- Multiple alternative
- Nested structures

H. Loops **3 hours**

- While loop
- Do-while loop
- For loop
- Counting loop
- Priming read loop

I. Programming language form **2 hours**

- Syntax
- Structured code
- Documentation
- Case sensitivity

J. Advanced formatted I/O including file access **4 hours**

- Formatted input and output
- File I/O
- Sequential file processing

K. Testing and Debugging Techniques **1 hour**

L. Functions **3 hours**

- Argument passing
- Returning results

- Recursion
- Testing a program system

M. Introduction to the Object Oriented Approach **1 hour**

N. Class Examinations **3 hours**

VII. Course Software Requirements

Having the following application software and usage skills are required to successfully complete this course:

- Internet connection and a World Wide Web Browser such as Internet Explorer 6.0 or higher so you can access your Moodle account.
- Electronic Mail using IUP's I-mail system.
- Microsoft Office 2007 or higher. Particularly, you need Word and PowerPoint applications to view various course contents. Free viewers such as "MS PowerPoint Viewer 2007" may be downloaded from Microsoft Web Site http://www.microsoft.com.
- Adobe Reader is needed to open PDF files. It can be freely downloaded from http://www.adobe.com/. Also, Adobe Flash Player can be downloaded from the same site if needed.
- Microsoft Visual Studio .NET 2005 or higher. This Integrated Development Environment (IDE) software can be obtained at no cost on DVDs from the Department of Computer Science through its Academic Alliance Agreement with Microsoft. For those students who neither have this software nor can borrow copies of the DVDs, they can download a freeware version of this program called "Microsoft Visual C++ 2008 Express Edition" from the following link http://www.microsoft.com/express/download/. This freeware only supports the C++ language which is sufficient for the purpose of this course.

VERY Important Notice:

It is understandable that almost all of you will be using your own personal machines or any other computing resources outside IUP. Given that, it is your own responsibility to ensure that those computers are properly equipped with ALL software programs mentioned above and these programs are properly operating. The bottom line is simply that: ALL your assignments and submitted work rely on the effective use of all the software specified above. The Computer Science Department Open Lab located at 112B Stright Hall has all the required software installed in case you might need to use a lab at IUP.

VIII. Class Attendance

Class attendance will be determined by meeting deadlines for assignments. Missing assignment deadlines will count as one class session missed. The allotted absence is three classes. If there is an emergency, notify me immediately. **TEN** points will be deducted from the final points earned for assignments for each day missed beyond

the allotted three classes. Missing half of the course will result in an automatic F in the course. Not taking the final exam will also be an automatic failure for the course. Participation on other synchronous activities might also be required.

IX. Evaluation Method

Grading Scale
A >= 90% , B >= 80% , C >= 70%, D>= 60%, F < 60%

Approximate Point Distribution	**% of Grade**
Two Examinations (Midterm and Final)	30%
Three Quizzes	15%
Programming Projects (between 6-9 projects)	35%
Programming Exercise Completions/CodeLab	10%
Discussion Forum Activities	10%

Note: Other activities as indicated in the course Site on Moodle might be requested from students and these might worth portion of the course grade.

X. Course Expectations/Policies

The instructor will do his best to respond to students promptly at the same time students must adhere to the following list of important expectations. This list is not meant to be comprehensive; therefore, other common sense expectations should also be considered. Each student MUST:

1. Access Moodle and IUP email frequently (daily basis is strongly recommended).
2. Be very familiar with the use of various Moodle tools.
3. Know the due dates of all assignments and submit them into Moodle on time. Note, the Moodle system will not allow any late submission.
4. Work at her/his own pace as long as s/he is satisfying the requirements and deadlines set in the tentative course schedule below and/or posted on Moodle.
5. Work INDIVIDUALLY through the listed work. Receiving any external assistance is not permitted unless explicitly approved by the instructor.
6. Submit all work through Moodle. Any work that is emailed to me and NOT submitted through Moodle will receive **NO** grade, unless the instructor has been notified before hand and the student presented an acceptable documented excuse.
7. Do not (under any circumstances) submit an assignment, if the model solution for that assignment has been posted.
8. Distribute his/her time devoted to class work in order to fulfill the 42 hour class time requirement. Use the tentative course schedule table below and/or posted on Moodle as a guideline. Time management is so crucial in an online course.
9. Generally, any deadline stated in this syllabus should be at **11:55 PM EST on that particular day**.

XI. Course Content and Activities

This course presents technical content and employs a variety of activities with the central objective in mind that the learning outcomes of this on-line course should be equivalent to those achieved if the course is offered as a face-to-face one. One other important reason for the use of the following wide spectrum of tools and activities is to ensure that the course will foster a community of learners in which all the students and the instructor are involved in class activities and all are contributors to the learning process. It is important to note that, research has proven that the quality of students' activities has a higher weight than the delivery mechanism [17]. Consequently, the expectation of achieving the equivalent learning outcomes as in face-to-face offering is a reality as long as the on-line course is carefully designed to ensure adequate levels of student involvements in high quality activities. The tools and activities proposed can be generally divided in Synchronous and Asynchronous ones. Below is a brief explanation of them along with a rough categorization of various tools/content/activities. Also, a brief description of how those are carefully employed to serve achieving the above-mentioned course outcomes is presented.

Course Content: The following Materials will be available through Moodle. Each student will be required to logon to his/her account to access this content.

- **PowerPoint Presentations:** All presentations for the covered chapters of the adopted textbook will be posted on Moodle. These will serve as a high-level documentation of the text covering the necessary programming topics needed for students to master various programming skills. These PowerPoint presentations should be considered as chapters' outlines and by no means can substitute reading the chapters from the book.
- **Class Notes:** Although the presentations above will cover all the necessary topics, making class notes available is so essential in such a fully on-line environment. Class notes abstract the long-term experience of the instructor presented in an easy-to-understand and legible style to students.
- **Source Code for all Programs in the Required Textbook**: Learning how to program computers relies mainly in the ability to think abstractly and to develop a general high-level algorithmic solution to the given problem. The more students get exposed to programming examples that handle various problems the more these skills get improved and polished. Posting these large numbers of examples will definitely help students with this respect.
- **Stored Video Lectures:** An optional content that might be used is to record class lectures as video files and making them available through the IUP Streaming Media Service.

Asynchronous Activities/Tools: Students will be required to perform the following activities and work at his/her own pace to ensure full understanding of the covered topics.

- **Reading Assignments:** all students will be asked to promptly read the chapters of the adopted textbook based on the tentative course schedule. Two main support

reading resources that will also be posted on Moddle are the PowerPoint Presentations and Class Notes as indicated in Course Content above.

- **Laboratory Exercises**: A comprehensive set of Laboratory Exercises are to be developed to cover all programming topics listed in the detailed course outline above. For students to become good programmers, they have to practice developing programs themselves. It is like any sport that requires mastering a set of skills and those cannot be learnt without practicing these activities over and over again. These labs are to be carefully designed with the nature of this course in mind to ensure that students are able to individually go through them and achieve the intended learning outcomes. All students are requested to complete all labs and submit the completed work on-line.
- **CodeLab Web-Based Tool:** In addition to the set of Laboratory Exercises above, all students will be asked to subscribe to an on-line tool called CodeLab. The semester-based subscription cost for a student is very reasonable and is around $20. This tool is an excellent environment for teaching students how to develop programs using the C++ language. It helps students to get very familiar with the syntax of various language constructs in addition to instilling into them various essential programming skills. **CodeLab provides students with instantaneous intelligent feedback to all their activities, a crucial feature that will compensate for the absence of physical contact with the instructor.** Successful completion of all CodeLab exercises is part of the course grade as detailed in the Evaluation Method. The instructor will monitor students' progress on these activities and grades will be assigned in proportion to how much of these exercises were fully performed. CodeLab provides exercises specifically addressing the adopted text while giving instructor the ability to customize them.
- **Discussion Forum Tool:** This asynchronous tool represents, if used effectively, one of the powerful techniques to ensure students' involvements in the class and can be used to cultivate a community of learners in which all members of the class are active contributors to the learning process. This tool also provides the main vehicle for teacher-student and student-student interactions. Several discussion forum topics are designed to carefully ensure maximum students' involvement and participation in these activities. That participation will be part of the course grade as indicated in the Evaluation Method. Discussions board activities will be graded based on quality of participation and posted contents. Some of the factors considered in grading various postings include correctness, style, depth, and comprehensiveness. Some of the discussion topics are briefly described below:
  - **Main**: Students are asked to post all their questions and concerns under this topic. As the topic name indicates, the Main discussion topic is meant for general discussions and students are encouraged to reply to each other *(as long as they follow the guidelines listed in the Online Etiquette section below)* and post their questions and concerns under this topic. Also, my replies to your general questions will be under this topic.
  - **Course Overview**: Students might opt to post a query to this topic, if they have any question on the course syllabus. It is not mandatory to participate in

this discussion forum topic. This will take place only during the first week of classes then this topic will be locked.

- **Introductions**: Every student is required to participate in this topic. This requires each student to post four facts about her/him-self to this topic of the discussion forum. This exercise is graded. To get a grade, students must post these facts by the end of the starting week of the course (by 11:55 PM EST).
- **Other Programming Topics:** All students will be constantly required to participate in various other programming discussion forum topics such as Decision Making, Looping, Input/Output, Arrays, etc. Please check the tentative course schedule below for a complete list of these topics. In each one of these timed and planned activities, the instructor will first post several questions addressing the main concepts of the topic under discussion. All students will be asked to provide answers to these questions and comment on at least two other postings by other students regarding this activity. The instructor will continuously monitor the discussion and will ensure that all are involved and will steer the discussion in the proper direction.

- **IUP Email**: Although IUP Email (I-mail) can be used as another medium to communicate with the instructor, I strongly discourage all students to send course-related questions/concerns to me by email (unless it is absolutely necessary). All such questions/concerns need to be posted on the appropriate discussion forum topics, see above. ***Note, emails from non-IUP accounts will NOT be considered.***
- **Programming Projects**: Most of the above activities are geared towards teaching the students how to program. This activity has the same objective but at the same time will be a considerable factor in assessing the level of acquired programming skills for students. Carefully chosen programming projects will be assigned and students will be asked to write, compile, and run their own C++ programs to solve the given problem. All produced programs along with other related documents must be uploaded to Moodle before the project deadline.

Synchronous Activities/Tools: in addition to the above asynchronous activities and content, the following activities will be used to improve the quality of the on-line offering of this programming course.

- **On-line Chatting:** A number of chat rooms will be set up for students and instructor to get on-line together and discuss course related issues. The time for these chat sessions will be chosen to fit the majority of participants.
- **Telephone Calls**: The instructor will be available to answer any questions or concerns via phone during the online office hours listed at the beginning of this syllabus.
- **Live Classroom (LC)**: It is a tool that provides for virtual classroom where students and the instructor can communicate synchronously by sending audio, video, annotation, and other streams to all participants. This system also allows for sharing applications among all members of the class. LC can be accessed as one of the Moodle Activities. **It is a powerful Web-based application that significantly compensate for the lack of face-to-face interaction.** Suitable use of this tool includes offering review sessions before exams or in those cases that

the instructor deems appropriate. Because it is a synchronous activity, the time for these sessions will be chosen to fit the majority of participants.

Assessment Tools: The following tools will be used to assess student overall attainment of the intended learning outcomes of this course.

- **Quizzes:** After reading three to four chapters from the adopted textbook, students are expected to take a quiz on these read chapters. All Quizzes are available on Moodle and are to be taken online, submitted online, and must be taken during the availability period with **NO EXCEPTIONS. NO make ups for missed Quizzes or Exams will be offered.**
- **Exams:** There will be two Exams (the equivalent of Midterm and Final). All rules applied to quizzes also apply to Exams as discussed in the above item. The Final exam will be accumulative and covers all chapters. Please be advised that in addition to the limited availability period for both Quizzes and Exams, they are to be taken in very limited durations. Once exam duration is elapsed, you can neither change your answers nor answer new questions. **In all cases, you must submit the exam/quiz after finishing for it to be graded.**
- **Programming Project**: As described in asynchronous activities, submitted programming project will be graded and be part of the overall course grade.
- **CodeLab Exercises**: Successful completion of assigned CodeLab activities is also part of the assessment method as described before.
- **Discussion Forum Activities**: The quality of participation in such discussions is used as an assessment tool as it is part of the course grades.

XII. Tentative Course Schedule

Below is a table listing a tentative course schedule and other information related to course activities. **In case of differences between the information listed in this tentative schedule or generally listed anywhere in this syllabus and whatever is posted on Moodle, the information posted on Moodle should supersede what is in here as it will be the most up-to-date one.**

Week	Reading	Programming project	Labs	CodeLab Activity	Quizzes /Exams	Discussion Forum Activities
1	Chap 1	Project 1		Complete CodeLab ex CH1		**Introductions/ Course overview** topics are DUE
2	Chap 2		Lab1			
3	Chap 2/3	Project 2		Complete CodeLab ex CH2		**First Program** discussion forum topic is DUE
4	Chap 3		Lab2	Complete CodeLab ex CH3	Quiz 1 (Chap 1-3)	
5	Chap 4	Project 3		Complete CodeLab ex CH4		**Input/Output** discussion forum topic is DUE

6	Chap 5		Lab3			
7	Chap 5/6	Project 4		Complete CodeLab ex CH5	Midterm (Chap 1-6)	
8	Chap 6/7		Lab4	Complete CodeLab ex CH6		**Branching /Looping** discussion forum topic is DUE
9	Chap 11	Project 5		Complete CodeLab ex CH7	Quiz 2 (Chap 4-7)	
10	Chap 11	Project 6	Lab5	Complete CodeLab ex CH11		**Arrays** discussion forum topic is DUE
11	Chap 8			Complete CodeLab ex CH8		
12	Chap 9	Project 7	Lab6	Complete CodeLab ex CH9		**Functions/Scope** discussion forum topic is DUE
13	Chap 10			Complete CodeLab ex CH10	Quiz 3 (Chap 8-11)	
14	Chap 13	Project 8	Lab7	Complete CodeLab ex CH13		**Structures/ Recursion** discussion forum topic is DUE
15	**Final Exam (Chap 1-11 & 13)**					

XIII. Online Etiquette

Diversity has many manifestations, including diversity of thought, opinion, and values. You are encouraged to be respectful of that diversity and to refrain from inappropriate commentary. Should such inappropriate comments occur, we will intervene as needed. You as well as faculty will be guided by common sense and basic etiquette. The following are good guidelines to follow:

a. Never post, transmit, promote, or distribute content that is known to be illegal.
b. Never post harassing, threatening, or embarrassing comments.
c. If you disagree with someone, respond to the subject, not to the person.
d. Never post content that is harmful or abusive; racially, ethnically or religiously offensive; vulgar, sexually explicit or otherwise potentially offensive to readers.

XIV. Academic Integrity Policy

Students are expected to uphold the school's standard of conduct related to academic honesty. Students assume full responsibility for the content and integrity of the academic work they submit. The guiding principle of academic integrity shall be that a student's submitted work, examinations, reports, and projects must be that of the student's own work. Students shall be guilty of violating the honor code if they:

a. Represent the work of others as their own.
b. Use or obtain unauthorized assistance in any academic work.
c. Give unauthorized assistance to other students.
d. Modify, without instructor approval, an examination, paper, record, or report for the purpose of obtaining additional credit.
e. Misrepresent the content of submitted work.
f. Submit work done in other classes.

The penalty for violating the honor code is severe. Any student violating the honor code is subject to receive a failing grade for the course and will be reported to the Office of Student Affairs. If a student is unclear about whether a particular situation may constitute an honor code violation, the student should communicate his/her concerns with the instructor to discuss the situation.

XV. Disabilities Policy

In compliance with the Americans with Disabilities Act (ADA), all qualified students enrolled in this course are entitled to "reasonable accommodations." Please notify the instructor during the first week of class of any accommodations needed for the course.

XVI. Bibliography

1. Breedlove, T. and Albert, R., C++: An Active Learning Approach, Jones & Bartlett, 2009.
2. Chou, P. and Chen, H., "Engagement in Online Collaborative Learning: A case Study Using Web 2.0 Tool," Journal of Online Learning and Teaching, Vol. 4, No. 4, pp. 574-582, 2008.
3. Deitel, P. and Deitel, H., C++ How to Program, 7th Edition, Prentice Hall, 2010.
4. Delving, M., Coenen, F., and Leng, P., "Teaching Java in an On-Line Degree Programme: A Case Study,' Proceedings of the Sixth Java & the Internet in the Computing Curriculum Conference, 2002. Last accessed on 12/5/2009 from http://www.ics.heacademy.ac.uk/events/presentations/515_devlin.doc.
5. Dongsong, Z., Zhao, J., Zhou, L., and Nunamaker, J., "Can E-learning Replace Classroom Learning?" Communications of the ACM, Vol. 47, No. 5, pp. 75-79, 2004.
6. El-Sheikh, E., "Techniques for Engaging Students in an Online Computer Programming Course," Journal of Systemics, Cybernetics and Informatics, Vol. 7, No. 1, pp. 1-12, 2009.
7. Farrell, J., Object-Oriented Programming Using C++, 4th Edition, Course Technology, 2009.
8. Friedman, F. and Koffman, E., Problem Solving, Abstraction & Design Using C++, 5th Edition, Addison-Wesley, 2007.
9. Gaddis, T., Starting Out with C++ Brief: From Control Structures through Objects, 6th Edition, Addison-Wesley, 2010.

10. Gill, G., Introduction to Programming Using VISUAL C++ .NET, John Wiley & Sons, 2005.
11. Horstmann, C. and Budd, T., Big C++, 2nd Edition, John Wiley & Sons, 2008.
12. Horton, I., Beginning Visual C++ 2008, Wrox, 2008.
13. Karsten, R., Kaparthi, S., and Roth, R., "Teaching Programming via the Web: A Time-Tested Methodology," Journal of College Teaching Methods & Styles, Vol. 1, No. 3, pp. 73-79, 2005.
14. Lafore, R., Object-Oriented Programming in C++, 4th Edition, Sams, 2002.
15. Lee, M., C++ Programming for the Absolute Beginner, 2nd Edition, Course Technology, 2009.
16. Liang, D., Introduction to Programming with C++, 2nd Edition, Prentice Hall, 2010.
17. Maki, R. and Maki, W., Online Courses. In Durso, F., Dumais, S., Lewandowsky, S. and Perfect, T., Handbook of Applied Cognition, 2nd Edition, pp. 527-552, John Wiley & Sons, 2007.
18. Malik, D., C++ Programming: From Problem Analysis to Program Design, 5th Edition, Course Technology, 2010.
19. Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., Delvin, M., and Paterson, J., "A survey of Literature on Teaching of Introductory Programming," Proceedings of the Annual Joint Conference Integrating Technology into Computer Science Education, pp. 204-223, 2007.
20. Savitch, W., Absolute C++, 4th Edition, Addison-Wesley, 2010.
21. Stephens, D., Diggins, C., Turkanis, J., and Cogswell, J., C++ Cookbook, O'Reilly, 2005.
22. Thomas, R., "Experiences Teaching C++ Online," Journal of Computing Sciences in Colleges, Vol. 15, No. 5, pp. 214-222, 2000.
23. Truong, N., Bancroft, P., and Roe, P., "ELP - A Web Environment for Learning to Program," Proceedings of the 19th Annual Conference of the Australasian Society for Computers in Learning and Tertiary Education, 2002.
24. Unuakhalu, M. and Kumar, A., "Challenges of Teaching online Programming Courses," Proceedings of International Conference on Advances in Interdisciplinary Statistics and Combinatorics, 2007.
25. Wang, F., Fong, j., Choy, M., and Wong, T., "Blended Teaching and Learning of Computer Programming," In Advances in Web Based Learning, Springer Verlag, pp. 606-617, 2008.

COSC 110: Problem Solving and Structured Programming
Sample Lesson: For Loops
Distance Education Course

Instructor: Waleed Farag, Ph.D.

The following lesson will briefly presents how a number of proposed activities in the online syllabus of record can be used in meeting one of the course objectives. In particular, this lesson will elaborate on the use of CodeLab by students in order to fully understand how the looping construct, *for loop*, can be used to repeat program statements. **This lesson specifically addresses course objective #9 (Use these programming elements: …. , loop structures, …).**

Objectives:

After completing this lesson, the student will be able to:

- Describe the need for repetition in computer programs
- Differentiate between count-based and event-based loops
- Use proper logic for *for loop construct* in the student's developed applications
- Identify various syntax issues while dealing with the *for loop construct* and describe how these can be fixed.

Activities:

Students are asked to conduct the following:

1. Read Chapter 9 from the adopted textbook. Also, carefully study the class notes posted on-line.
2. Investigate and experiment with all code examples posted on-line and focused on *for loops*. Examples of these are BMIWFor.cpp & CharCounts.cpp which can be found in Chap 9.
3. Complete all CodeLab Exercises on Loops (Chap 9).
4. Answer the *for loop* questions posted on the discussion forum and comment on two postings of other students on the same subject.

Assessment:

The following is a set of assessment tools to be used in evaluating students on this lesson:

- Completion of all CodeLab exercises.
- A programming project focused primarily on *for loops* will be given to students then graded.
- Student's postings on the discussion forum will be assessed based on the quality of postings.
- A Quiz on this concept will be offered to assess the student overall understanding of it.

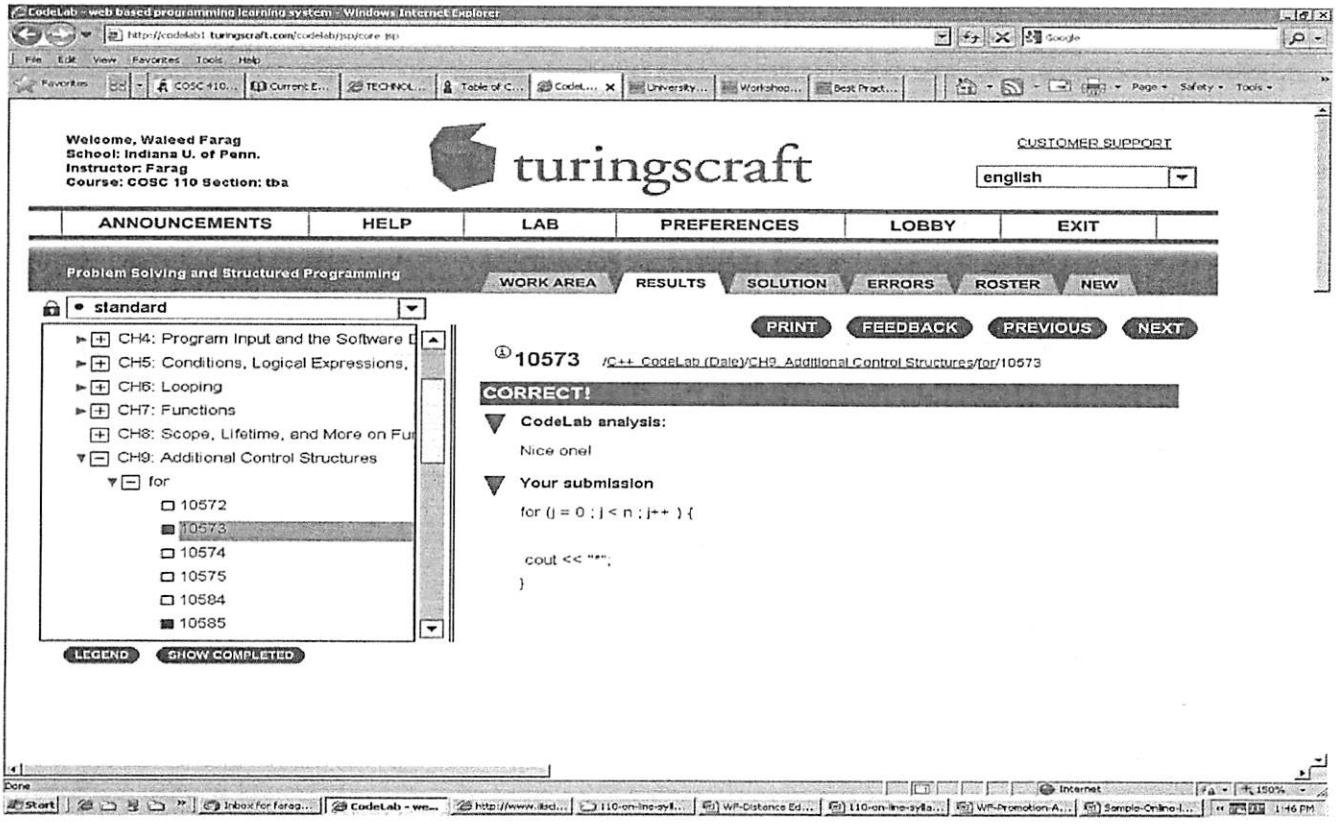
The remaining of this lesson will mainly focus on how CodeLab, Activity #3 above, can be used to achieve various lesson objectives. Two Scenarios are given below one in which the student performs the activity properly and in this case the system reports that. The other one shows the intelligent and instantaneous feedback provided by the system in case of the answer was not correct.

Student will perform the following:

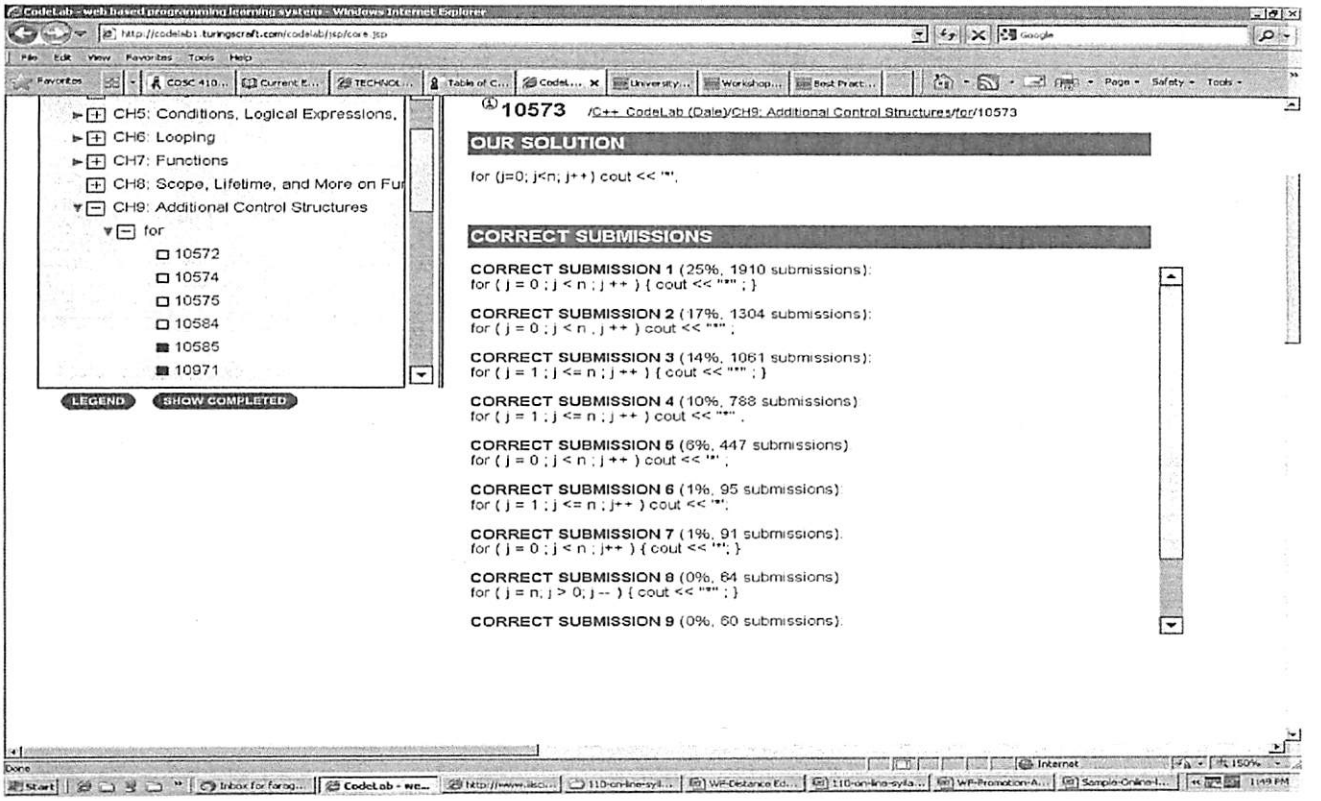
1. Logon to the CodeLab Web-based system using the below link.
   http://codelab1.turingscraft.com/codelab/jsp/login1.jsp
2. Click on the tab LAB then click on WORK AREA.
3. Select Chapter 9 (**CH9**) from the left Panel and select **for** then exercise # 10573.
4. Read the problem statement, write the code in the provided work area then click Submit.

The following Screen shot shows the status before submitting the code. The code is correct in this case.

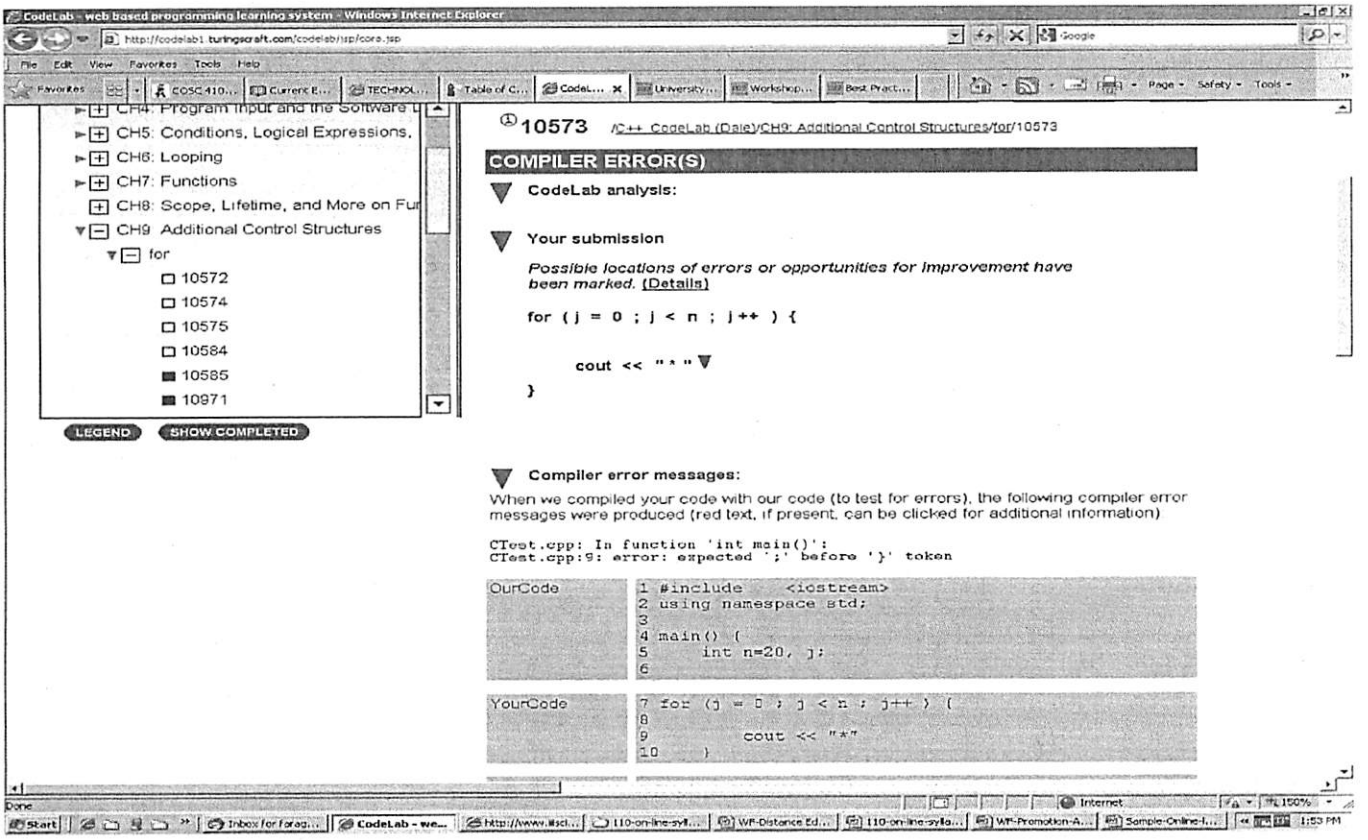
The following screen shot shows the system's response in case of correct answer.



The system also provides excellent feedback regarding other possible valid solutions and the percentage of submissions as indicated in the following screen shot.



The other scenario is when the student's answer is wrong. Below is another snapshot showing the system's response in such case.



Note how the system intelligently identifies the error location (with a red inverted triangle) and gives the student very helpful feedback on how to fix the issue. The system also displays how the student's written code was included into the overall program generated by the system in order to compile the student code. This gives the student a much better understanding of the big picture. Students can now fix the problem and resubmit the code which will result in displaying a snapshot similar to the second one above.

This will present a very effective and instantaneous feedback to students. Each student can practice at her/his pace and as long as the exercise is completed successfully, full credit will be granted.

(End of Lesson)