Appendix I (Part 1) (rev 8/2017)
Program/Discipline Assessment Results

Program/Discipline Assessment Results Form for
Computer and Information Science (050.3);
an Associate of Science degree program

Academic Year 2017

Specific Program/Discipline Objective to be Assessed	Apply core principles and practices of computing (Program completion objective #1)

Number of faculty who participated	2
Number of faculty who could have participated	9
Number of students participating	54
Face-to-face sections	4
On-line sections	0

Summarized Results

Updated as of 11 December 2017 at 16:00h

Tabulation	Count	Total	Percent
Task 1 (Direct measure): Translating simple logic flow charts into Java code that compiles			
Students achieving score 4	28	54	**52%**
Students achieving score 3	11	54	**20%**
Students achieving score 3 or greater	***39***	***54***	***72%***
Students scoring 2 or lower	13	54	**24%**
Task 2 (Direct): "Paper Compiling" a Java application with 1 class, a main() method , and a doMath() method			
Students achieving score 4	31	54	**57%**
Students achieving score 3	6	54	**11%**
Students achieving score 3 or greater	***37***	***54***	***69%***
Students scoring 2 or lower	17	54	**31%**
Task Average			
Average score 3 or greater	37	54	**69%**
Average score less than 3	17	54	**31%**

Results described

	Direct Measure ***Flow Charts → Java Code***	**Second (Direct) Measure:** ***"Paper compiling" basic decision logic***
Results **Face-to-face sections**	75% of students performed earned a rubric score of 3 or greater	75% of students performed earned a rubric score of 3 or greater
On-line sections	No online sections participated	No online sections participated
According to your results, is the objective being achieved? **(provide explanation)**	**YES!** The criteria of 70% of students scoring a 3 or greater was logged in the accompanying Appendix E prior to the administration of the assessment. According to this technical measure, this cohort of students can collectively be thought to be achieving this objective. Given that a preponderance of students in CIT-111 sections across the College are in an exploratory phase in their education path, and most have not declared enrollment in CIS 050.3 associates program, these are solid results since they tested students at a rigorous level.	**Almost!** 69% of students who completed task two scored a 3 or a 4, meaning proficient or advanced. This is 1% below the criteria originally set for meeting competence of 70%. It is notable, however, that of the 69% who scored 3 or 4, 57% of those scored a 4, suggesting about half of the students are performing at a level of competence and confidence.
Strengths and Weaknesses of student learning uncovered during this assessment	**Strengths:** Nearly each student who completed the instrument created an application which made a solid attempt at implementing a two-branch decision logic. This reveals command of the basic components of a Java application and the essential control-of-flow mechanism: if/else **Weaknesses:** A quarter of students who took the exam were unable to correctly translate decision criteria written in English into straightforward symbolic logic which compiles without error. Errors ranged from improperly configured comparison operators to incorrectly chaining	**Strengths:** 37 of the 54 students who completed the assessment were capable of tracing the life of two integer values as they are passed into a method outside of main(), computationally combined, and returned to main() to be incremented and printed. This represents command of the fundamental communication method in Java: method calls with input parameters and output types. **Weaknesses:** 1) Students consistently did not demonstrate useful note taking and annotation of the code that would support "paper compiling" more complicated logic. For many

	if/else blocks	students, their tracing of the values of a and b through the application was done entirely in their head, which is extremely error prone. 2) A handful of students were not able to correctly apply basic numeric operations on two variable values with an enforced order of operation. Lacking this foundational math skill limits students' ability to correctly implement a whole range of essential algorithms in computer science.
Action(s) to be taken by faculty for improvement of learning	Create a series of learning exercises which **isolate the skill of translating English program logic specification into Java code** which uses binary and unary operators. These exercises should begin with **logic only requiring a single if/else block** with simple logical operators. More advanced exercises should involve "branched and chained" if/else statements that require parsing a multi-step decision process in English into Java code	1) Create an **algebra primer and refresher exercise** set that includes challenges for students with the most rudimentary math skills as well as students who are comfortable (fluent) in algebraic operations. This should be a "spiraling" process such that students gain repetitive practice with these math skills while also practicing Java coding. 2) Create and implement mini-lessons on **how to make notes and diagrams to use while working through "paper compiling"** exercises that require careful tracing of values through a series of transformations, method calls, and assignment operations.

Submitted/prepared by: Eric Darsow, Faculty Instructor, Computer Department, North Campus, Community College of Allegheny County. 412.894.3020, edarsow@ccac.edu

Appendix I (Part 2) – Follow-ups After Taking Action

Evidence that assessment results were used to inform decisions regarding instruction, curriculum and resource allocation.

Actions Taken	Evidence of Improved Learning	Use of results
Click here to enter text.	Click here to enter text.	☐ instruction ☐ curriculum ☐ resource allocation
Click here to enter text.	Click here to enter text.	☐ instruction ☐ curriculum ☐ resource allocation
Click here to enter text.	Click here to enter text.	☐ instruction ☐ curriculum ☐ resource allocation
Click here to enter text.	Click here to enter text.	☐ instruction ☐ curriculum ☐ resource allocation

Submitted/prepared by: Click here to enter text.

Examples for Actions and Use of Results

Instruction:

- Experiment with alternative teaching formats (e.g., problem-based learning)
- Provide additional in-class or out-of-class assignments
- Require an early non-graded first draft and provide feedback

Curriculum:

- Revamp departmental curriculum to require more oral and written communication
- Examine prerequisite courses, change course sequence
- Discuss assessment findings with colleagues in departments with similar goals

Resource Allocation

- Provide additional resources for students (DVD's, more lab time, reference materials)
- Require use of existing campus resources (Learning Commons, Job Placement & Career Services, etc.)
- Use assessment results to justify budget requests