Table of Contents

Assessment outcomes for reporting

Outcome
5.2.1 Describes and applies problem-solving processes when creating solutions
5.2.3 Critically analyses decision-making processes in a range of information and software solutions
5.5.3 Describes and compares key roles and responsibilities of people in the field of information and software technology

Driving question:

In doing the NCSS Challenge, how have my knowledge and skills in designing and implementing coded solutions grown, AND what does this mean to me now and into the future?

PBL Summary

Discover	Create	Share
• Designing and communicating solutions (algorithm representation) • Coding solutions in python • Fundamental concepts in programming • Careers in s/w development	• Weekly Evaluations, critically unpacking computational thinking as a problem-solving process • Comparison of algorithms • Reflection on growth in SDP	• Reflections will be featured post in Grok Learning blog - this can be text, podcast, infographic, and/or video

Assessment plan

Final products Products in **< > brackets** are summative, i.e. assessed for academic reporting	**Learning outcomes/ targets**	**Checkpoints/Formative Assessments**	**Instructional strategies / resources**
<Weekly evaluation> Evaluate specific problems personally chosen from the NCSS challenge. *Marking (5 * 4 = 20)* There are 5 possible entries and best 4 will be counted for marking. **<NCSS completion rate>** Percentage of problems solved, regardless of NCSS scores. *Marking (5 * 1 = 5)* 80% + = 5 65% + = 4 50% + = 3 25% + = 2 1% + = 1	5.2.1 Describes and applies problem-solving processes when creating solutions	**Evaluation class notebook** Check every week to ensure students are completing this **NCSS Teacher dashboard**	- Keyword notes on class OneNote notebook - NCSS Challenge scaffold on class OneNote notebook - NCSS challenge - Refer to chapters 3 and 15 in textbook - Teacher model use of CASE tools for flowcharts : Lucid Chart and draw.io - 1-1 coaching as required
<Research on Careers> Investigate career options, focusing on two in particular. Investigate means to plan, inquire into and draw conclusions about	5.5.3 Describes and compares key roles and responsibilities of people in the field of information and software technology	**Research draft**	- Writer's workshop – writing reflections, explanations, justification, compare and contrast - Keyword notes on class OneNote notebook - NCSS Challenge scaffold on class OneNote notebook

Final products Products in **< > brackets** are summative, i.e. assessed for academic reporting	**Learning outcomes/ targets**	**Checkpoints/Formative Assessments**	**Instructional strategies / resources**
Marking (3 + 4 + 3 = 10) Marks correspond to job descriptions, job comparisons, and conclusion			- Refer to chapters 3 and 15 in textbook - Questioning techniques, whole class discussions - 1-1 coaching as required
<Reflection on growth> Using Weekly Evaluation, reflect on growth in programming skills and conceptual understanding including technical language. Align this with research on careers to define implications on current status as student and preferred futures - whether or not it involves s/w development. *Marking (5 * 2 = 10)* 5 marks each on how well growth and impact is explained. <**Algorithm comparison**> Find a problem from week 3 onwards where own solution differs from that of a peer, or teacher, or Grok's (ask teacher). Differences could be in logic, control structures, data types, data structures, and programming conventions - naming variables, comments, overall structure. List the differences. Explain why own solution still works, regardless.	5.2.3 Critically analyses decision-making processes in a range of information and software solutions	**NCSS Teacher dashboard** **Reflection draft** **Choices/drafts for algorithm comparison**	- Keyword notes on class OneNote notebook - NCSS Challenge scaffold on class OneNote notebook - NCSS challenge - Refer to chapters 3 and 15 in textbook - Direct instruction of algorithm representation - LEGO algorithm challenges - Algorithm representation practice, using NCSS challenge problems - 1-1 coaching as required

Final products Products in **< > brackets** are summative, i.e. assessed for academic reporting	**Learning outcomes/ targets**	**Checkpoints/Formative Assessments**	**Instructional strategies / resources**
Marking (2 + 3 = 5) Marks correspond to outline of differences and explanation. Bonus 2 marks for significant difference in logic (most likely for harder problems), acknowledging complexity in differences, i.e. This would likely involve differences in control structures as well as data structures as well. Total marks awarded will not exceed assessment total of 50.			

Weekly Evaluation

Problem title:

Name the NCSS problem you found interesting and/or enjoyed this week. Just type in the title; no need to copy the question.

Level: Beginner / Intermediate / Advanced

Ratings:

Range of 1 to 10; delete excess icons. Effort is related to difficulty of the problem but not necessarily, e.g. You knew how to solve the problem (found it easy) but had to fix silly errors or needed to revise notes and syntax.

Effort rating	Enjoyment rating
🏅 🏅 🏅 🏅 🏅 🏅 🏅 🏅 🏅 🏅	❤ ❤ ❤ ❤ ❤ ❤ ❤ ❤ ❤ ❤

Problem solving process:

Refer to chapters 15.11 - 15.12 (Error detection and correction) in your textbook.

Problems encountered (Outline problems, see examples below)	**Problem-solving strategies** (Check as many as appropriate)
Syntax missed ')' in print statement • **Logic** missed a question requirement • **Run-time** Used the wrong data type for input variable •	□ Read question carefully □ Plan solution carefully: IPO, pseudocode, flowchart □ Good programming conventions o Logical structure o Meaningful variable and function names o Internal documentation □ Read error messages carefully □ Revise previous NCSS lesson notes □ Trial and error □ Ask peers □ Ask teacher □ Ask NCSS tutor □ Ask others, e.g. parent, sibling □ Ask Google □ Code walkthrough □ Desk-check □ Peer review of the program code □ Debugging output statements

	□ Thorough testing using a range of data including data that is deliberately wrong □ Stubs □ Flags □ Use a debugger, e.g. Python Tutor □ Other, please indicate

Algorithm representation in pseudocode and/or flowchart

Digitised text or drawing only. Flowcharts can be hand-drawn or use CASE tools, e.g. Lucid Chart and draw.io

Do not paste your coded solution here.

Research on Careers

Job Descriptions

Choose and *describe* 2 software development roles, including web development

Job comparisons

Compare and contrast chosen roles on certain features - you can add to the list below. Outline similarities, differences and link relevance or implications to own self.

Feature	**Similarities**	**Differences**	**Personal Relevance**
Job responsibilities			
Education and training			
Pay scale/range			
Career path			

Conclusion and Justification

Conclude which of the two jobs you prefer and *justify* why based on details above.

Reflection on growth

Using completed evaluation throughout the 5-week NCSS challenge, reflect on your growth as a software designer and programmer. Complement this with research on careers to write your response to the **Driving question**: *In doing the NCSS Challenge, how has my knowledge and skills in designing and implementing coded solutions grown, AND what does this mean to me now and into the future?*

There are two parts to this question, i.e.

- Show how you have grown (or not) in designing and implementing coded solutions
- Explain what this mean to you now and your future, linking to potential (or not) career in software development

Clear and explicit (hyper-)links to completed logs and careers research are considered more thorough and compelling.

Reflection can be text, podcast, infographic, and/or video format.

Reflection planner

Use this planner to help you structure your reflection. This page will not be marked and completion is recommended but optional.

Consider the following questions:

- Have you grown as a software designer and programmer as a result of completing the NCSS Challenge?
- If so, how much have you grown?
- How do you know that you have grown?

	Conceptual understanding including technical language	**Technical Skills**	**Key points / summary / link to career research**
S/W design	Compare week _ to week _ AND/OR Learned ___ on week _ **Impact**	Compare week _ to week _ AND/OR Learned ___ on week _ **Impact**	
S/W programming	Compare week _ to week _ AND/OR Learned ___ on week _ **Impact**	Compare week _ to week _ AND/OR Learned ___ on week _ **Impact**	

Conclusion:

- Overall, how have you grown in designing and implementing coded solutions?
- What does this mean to you now and your future?