(http://csmatters.org) C - 1

0bC - 0b1

Create - Application from Ideas

Unit Create Performance Task

Revision Date: Jul 18, 2016
Duration: 15 50-minute sessions

Lesson Summary

Pre-lesson Preparation

Students should have practice in the following areas before starting on the Create Performance Task:

- Designing a program using multiple-level algorithms (at different levels of abstraction) and data abstractions, and being able to identify how they are used.
- Creating a program with the following features:
  - Mathematical expressions.
  - Logical expressions.
  - Sequence, selection, and iteration.
  - Functions.
- Creating required submission documents, including:
  - Creating a video (narration optional) that illustrates the functionality of a program.
  - Saving program code as a PDF.
  - Marking segments of the code using ovals and squares to signify algorithms and the use of abstraction.

Summary

In the Create Performance Task (CPT), students bring ideas to life through software development. The CPT requires students to design and iteratively develop a program. The College Board encourages but does not require students to collaborate with a partner, and once students begin the development of the CPT, they may receive help only from the collaborative partner. While not requiring it, this lesson supports a collaborative effort. Even if collaboration is used, each student must independently complete a significant level of planning, designing, and developing the program. Students will have 12 hours of class time to complete, and submit:

- A video of the program running.
- Written responses about the program and the development process.
- All of their program code.

The program must:

1. Produce a result that cannot easily be accomplished without computing tools.
2. Demonstrate a variety of capabilities.
3. Include an algorithm that integrates other algorithms and mathematical or logical concepts.
4. Use abstraction (functions or data structures) to manage the complexity of the program.
5. Implement several language features and use the following concepts:
    - Mathematical expressions such as numbers, variables, and arithmetic operators.
    - Logical expressions using Boolean operators.
    - Decision or selection branches.
    - Iteration or looping.
    - Collections such as strings or lists.

This lesson plan provides a schedule for 15 50-minute sessions to meet the 12-hour in-class minimum required by the College Board.

Outcomes

- Students will design and implement a program to pursue a topic of personal interest or to solve a problem.
- Students will use functional or data abstraction to manage program complexity.
- Students will be able to implement a multiple-level algorithm.
- Students will be able to implement sequence, selection, iteration, and functions using arithmetic and logical expressions.

Overview

Session 1: Identify performance task requirements and choose a topic to develop.

Session 2: Design the program, identifying both multiple -evel algorithms and multiple abstractions used in their design. Students work with collaborative partners to verify the adequacy of their designs to meet the performance task requirements.

Session 3: Program Development: Code top-level functions.

Session 4: Program Development: Code second-level functions.

Session 5: Program Development Checkpoint: Assess progress and revise design and development strategies.

Session 6: Program Development: Continue revised function development.

Session 7: Program Development: Complete function development.

Session 8: Program Development Checkpoint: Assess progress and make final revision to development strategies.

Session 9: Program Development: Make final revisions to program code.

Session 10: Program Development: Complete program and plan video.

Session 11: Video Development: Record and optionally narrate video.

Session 12: Video Development: Assess video and make final revisions.

Session 13: Written Responses: Respond to prompts 2a, 2b, and 2c.

Session 14: Written Responses: Respond to prompts 2d and 2e.

Session 15: CPT Submission: Submit and share projects.

Learning Objectives

CSP Objectives

Essential Questions

- How does abstraction help us in writing programs, creating computational artifacts and solving problems?
- How are algorithms implemented and executed on computers and computational devices?
- How are programs developed to help people, organizations or society solve problems?
- How are programs used for creative expression, to satisfy personal curiosity or to create new knowledge?
- How do computer programs implement algorithms?
- How does abstraction make the development of computer programs possible?
- How do people develop and test computer programs?
- Which mathematical and logical concepts are fundamental to computer programming?

Teacher Resources

In the Lesson Resources Folder (Teacher Only)

- Create Performance Task Rubric as of November 2015

In the Lesson Resources Folder (Public)

- Preliminary Topic Selection Guide
- Create Performance Task Rubric Assignment
- Create Performance Task Directions as of November 2015
- Daily Progress Report
- Collaboration Agreement
- Program Development Plan

Possible Screen Capture/Recording Tool

VLC Media Player

Dowload: http://www.videolan.org/vlc/index.html (http://www.videolan.org/vlc/index.html)

Instructions for screen recording: http://www.wikihow.com/Screen-Capture-to-File-Using-VLC (http://www.wikihow.com/Screen-Capture-to-File-Using-VLC)

Lesson Plan

Teacher Guidelines for Involvement

Prior to administration of the Performance Task and during the planning stages:

Teachers MAY:

- Influence student selection of topics.
- Clarify directions for completion of the performance tasks.
- Remind students of submission requirements.
- Guide student understanding of the CPT.
- Designate collaborative partners
- **Influence** student selection of program topics (however, teachers may not require selection of particular topics for students).
- Assist students in creating a program development schedule.

Teachers MAY NOT:

- Choose topics for students.
- Develop, write, revise, or amend student work.
- Provide boilerplates.
- Allow students to submit practice tasks as actual performance tasks.
- Suggest or evaluate student responses to prompts.

During the Performance Task

Teachers MAY:

- Resolve hardware/technical problems.
- Manage student progress throughout the task.
- Assist students in managing their progress, including making assessments of student efforts and fulfillment of the development schedule
- Clarify directions for the performance tasks.
- Remind students of submission requirements.

Teachers MAY NOT:

- Write, revise, or amend student work.
- Provide boilerplates.
- Allow students to submit practice tasks as actual performance tasks.
- Provide programming support while students are developing the performance task. **Only code written by the student will be scored.**
- Provide feedback to students regarding the quality of the work until after the task has been finally submitted to the College Board.

Session 1

Objectives:

- Introduce the Create Performance Task (CPT).
- Students develop understanding of the CPT requirements.
- Students assess both the performance task requirements and its evaluation rubric.
- Students are designated partners and do preliminary topic selection.

Tell students: Today you will begin the Create Performance Task. In the CPT, you will be developing a program of your choice. The goal of this task is for you to develop a program to solve a problem, create an innovation, or express a personal interest.

Exercise 1: CPT Overview

Provide students with a copy of the Create Performance Task (AP Computer Science Principles Performance Task Create–Applications from Ideas) description from the College Board.

1. Students read the Overview section on page 1 and answer the three questions below. (2 min)
    a. How is the program topic determined?
    b. What three products are you to produce?
    c. How much time will you have to complete and submit the required products?
2. After students share with elbow partners, address any questions students have to this point. (3 min)
3. Students read the General Requirements section beginning on page 1 and answer the questions below. (5 min)
    a. What recommendation regarding collaboration is made?
    b. What are the five general requirements for this performance task?
    c. What does the video need to display?
    d. What are the written responses about?

Exercise 2: Program Requirements

1. Students read Program Requirements section. (1 min)
2. They then answer the questions below: (5 min)
    a. What must the program demonstrate?
    b. What must the program implement?
    c. What must the program produce?
    d. What five elements from the second sentence should the program contain?
    e. What must the program demonstrate about each of the following?
        - Mathematical and logical concepts
        - Implementation of an algorithm
        - Use of abstractions
3. After students share with elbow partners, address any questions that students have to this point. (3 min)

Exercise 3: Submission Requirements

1. Read part 1 of the Submission Requirements about the video and answer the following. (2 min)
    a. What three requirements does the video have to meet?
2. Read the first paragraph of part 2 about the Written Responses and answer the following. (2 min)
    a. What format is to be used for the written responses?
    b. How are the responses to be labeled?
    c. What is the maximum length of all written response combined?

Exercise 4: Written Response

Jigsaw the Program Purpose and Development section. Assign each of the written responses a-e to one of five groups. Each group answers the questions below. (2 min)

Written Response a

1. What must the response identify?
2. What must the response explain?
3. What is the approximate word limit?

Written Response b

1. What is focused on in this description?
2. In addition to describing the difficulties, what must be described about them?
3. At least one of the difficulties must have been addressed in what manner?
4. What is the approximate word limit?

Written Response c

1. What portion of the program code is to be copied in this section?
2. What is to be described?
3. What is the approximate word limit?

Written Response d

1. What portion of the program code is to be copied in this section?
2. What is to be explained?
3. What is the approximate word limit?

Written Response e

1. What is contained in this response?
2. What is to be marked with an oval?
3. What is to be marked with a rectangle?
4. When are comments are required?

Each group shares its results. (5 min)

Exercise 5:

Assign students collaborative partners. Distribute *Preliminary Topic Selection Guide*. Students decide on the topic they will address and the form of collaboration they will use. (10 min)

Homework:

Distribute the CPT rubric and CPT Rubric Assignment.

Session 2

Objectives:

- Students commit to a program topic.
- Students plan collaboration strategy.
- Students design the program with multiple-level algorithms and multiple abstractions.

- Students develop a formal schedule for program implementation.

Tell students:

Today we have four goals. We will commit to a program topic and develop a collaboration strategy, a program design, and an implementation schedule.

NOTE: Students may only receive help from the collaborative partner from this point on until the CPT documents are ready to be submitted. Teachers collect the topics, strategies, and designs in order to help manage student progress, but may not provide any feedback to students on content.

Exercise 1: Program Topic (5 min)

Meet with collaborative partners and discuss the CPT rubric assignment. Each individual submits programming topic and purpose.

Exercise 2: Collaboration Strategy (10 min)

Each partner group submits a plan for collaboration.

Exercise 3: Program Design (20 min)

Each individual submits a program design.

Exercise 4: Program Development Schedule (10 min)

Each individual submits a daily program development schedule.

Session 3: Program Development Day 1

Tell students: Today will be our first program design day.

Exercise: Students implement their program (48 min)

At the end of class:

- Students complete a daily progress report.
- Teachers will collect the progress report in order to monitor progress but may not provide any feedback to students on content. (2 min)

Session 4: Program Development Day 2

Tell students: Today will be our second program design day.

Exercise: Students implement their program (48 min)

At the end of class:

- Students complete a daily progress report.
- Teachers will collect the progress report in order to monitor progress but may not provide any feedback to students on content. (2 min)

Session 5: First Checkpoint

Tell students: Today we will evaluate program development progress and revise our program development schedules as needed. We will also discuss with partners the difficulties we encountered and the strategies we used to address them.

Note: Partners collaborating in their program development in the first session stages will switch to individual development at this point.

Exercise 1: Review Program Development Progress (5 min)

Each individual submits a revised schedule of program development,

Exercise 2: Managing Difficulties (5 min)

Partners discuss the major difficulties they have faced to this point and the strategies they used to overcome them.

Exercise 3: Students implement their program (38 min)

At the end of class:

- Students complete a daily progress report.
- Teachers will collect the progress report in order to monitor progress but may not provide any feedback to students on content. (2 min)

Session 6: Program Development Day 3

Exercise 1: Examining the Rubric (5 min)

Tell students: Before starting our research we will examine the rubric that readers will use when scoring your computational artifact and written response.

Exercise 2: Students implement their program (43 min)

At the end of class:

- Students complete a daily progress report.
- Teachers will collect the progress report in order to monitor progress but may not provide any feedback to students on content. (2 min)

Session 7: Program Development Day 4

Exercise: Students implement their program (48 min)

At the end of class:

- Students complete a daily progress report.
- Teachers will collect the progress report in order to monitor progress but may not provide any feedback to students on content. (2 min)

Session 8: Second Checkpoint

Tell students: Today we will evaluate program development progress and revise a program development schedule as needed. We will also once again discuss with partners the difficulties we encountered and the strategies we used to address them.

Note: Partners collaborating in their program development in the second session stages will switch to individual development at this point.

Exercise 1: Review Program Development Progress (5 min)

Each individual submits a revised schedule of program development.

Exercise 2: Managing Difficulties (5 min)

Partners discuss the major difficulties they have faced to this point and the strategies they used to overcome them.

Exercise 3: Students implement their program (38 min)

At the end of class:

- Students complete a daily progress report.
- Teachers will collect the progress report in order to monitor progress but may not provide any feedback to students on content. (2 min)

Session 9: Program Development Day 5

Tell students: Today will be our next to last program development day.

Exercise: Students implement their program (48 min)

At the end of class:

- Students complete a daily progress report.
- Teachers will collect the progress report in order to monitor progress but may not provide any feedback to students on content. (2 min)

Session 10: Program Development Day 6

Tell students: Today will be our last program development day.

Exercise: Students implement their program (48 min)

At the end of class:

- Students complete a daily progress report.
- Teachers will collect the progress report in order to monitor progress but may not provide any feedback to students on content. (2 min)

Session 11: Video Development Day 1

This session is the first of two scheduled to develop the video. It's possible to create the video in a single day, so if students need more time to work on their program implementation, it may be done here with video planning assigned as homework.

Tell students: Today will be our first of two days dedicated to producing the CPT video.

Exercise 1: Select Program Features (3 min)

Each individual selects features that demonstrate the programs overall purpose.

Exercise 2: Storyboard (10 min)

Students plan how to show the purpose of the program and the running of related features.

Exercise 3: Record clips for video (25 min)

Students record required clips.

Exercise 4: Video Clip Assessment (10 min)

Collaborative partners assess the video clips and recommend any additional clips or changes to existing clips.

Session 12: Video Development Day 2

Exercise 1: Video Completion (25 min)

Students complete the CPT video in mp4, wmv, avi or mov format.

Exercise 2: Draft Written Response 2a (25 min)

Students write the explanation of the programming language used, their program's purpose, and what the video illustrates.

Session 13: Written Responses: Program Development

Exercise: Students write responses to prompts 2b and 2c (50 min)

Session 14: Written Responses 2d and 2e

Exercise: Students write responses to prompts 2d and 2e (50 min)

Session 15: CPT Submission

Objectives:

- Submit all CPT documents by uploading them to the performance task submission web site.
- Present all computational innovations to the class.
- Complete a post task reflection.

Tell students: There are three goals for this session.

- Submit all three CPT documents by uploading them to the performance task submission web site.
- Present all programs to the class.
- Complete a post task reflection.

Exercise 1: Upload CPT documents (15 min)

Students upload both files.

Exercise 2: Create Program Presentations (15 min)

Students set up workstations to show their video and to give access to their program, Students perform a gallery walk to review the programs.

Exercise 3: Post Task Reflection (15 min)

Students respond to these two questions.

- What did you learn about application development?
- What advice would you give to students in next year's class?

Note: Teachers may want to record students making these comments to be used to introduce the project next year.

Evidence of Learning

Formative Assessment

Students may not receive assistance regarding the content of their CPT documents until after they have been submitted to the College Board.

Summative Assessment

Students may not receive assistance regarding the content of their CPT documents until after they have been submitted to the College Board. Once the documents have been submitted teachers should use the CPT rubric to assess performance task documents.

To convert the rubric score to a percentage teachers may want to use the following scale:

Program Development Progress: 50 pts

Over 10 days, students submit a project assessment, a program development plan and 6 daily progress reports and complete 2 program checkpoints. Weighted as 5 points for each day:

0 points no progress

1 point sporadic or minimal progress

2 points inconsistent progress

5 points consistent progress

Video Progress: 10 pts

5 points for each day

0 points no progress

1 point sporadic or minimal progress

2 points inconsistent progress

5 points consistent progress

Written Report Progress: 10 pts

5 points for each day

0 points no progress

1 point sporadic progress

2 points inconsistent progress

5 points consistent progress

Rubric score 30 pts

(http://www.umbc.edu/) 

(http://www.umd.edu/) 

(http://www.nsf.gov/)

Authored by: CS Matters in Maryland
Website: csmatters.org (http://csmatters.org)
Email: csmattersinmaryland@gmail.com (mailto:csmattersinmaryland@gmail.com)