

# CS 161A: Programming and Problem Solving I

## Assignment xx Algorithmic Design Document

---

*Make a copy before you begin (File -> Make a copy). Add the Assignment # above and complete the sections below BEFORE you begin to code. The sections will expand as you type. When you are finished, download this document as a PDF (File -> Download -> PDF) and submit to D2L.*

*This document contains an interactive checklist. To mark an item as complete, click on the box (the entire list will be highlighted), then right click (the clicked box will only be highlighted), and choose the checkmark.*

Planning your program before you start coding is part of the development process. In this document you will:

- ☐ Paste a screenshot of your zyBooks Challenge and Participation %
- ☐ Paste a screenshot of your assigned zyLabs completion
- ☐ Write a detailed description of your program, at least two complete sentences
- ☐ If applicable, design a sample run with test input and output
- ☐ Identify the program inputs and their data types
- ☐ Identify the program outputs and their data types
- ☐ Identify any calculations or formulas needed
- ☐ Write the algorithmic steps as pseudocode or a flowchart
- ☐ Tools for flowchart - [Draw.io](#) - [Diagrams.net](#)

### 1. zyBooks

---

Add your zyBooks screenshots for the % and assigned zyLabs completions below. Required percentages: all **assigned** zyLabs, Challenge Activity with at least 70%, and Participation Activity with at least 80%.

**Challenge and Participation % screenshot:**

Chrome File Edit View History Bookmarks Profiles Tab Window Help

learn.zybooks.com/zybook/CS161A-PCCCS161AWinter2022

zyBooks My library > CS 161A: Programming and Problem Solving 1 home

zyBooks catalog Help/FAQ Zakeriya Muhumed

Search content Search View content explorer

**Table of contents**  
About this material

zyLabs Challenge Participation

Chapter	zyLabs	Challenge	Participation
1. CS 161A: Introduction to C++	100%	100%	100%
1.1 Algorithms		No activities	
1.2 Flowcharts		No activities	
1.3 Algorithm Example: Calculating birth year			100%
1.4 Programming (general)			100%
1.5 Programming basics	100%		100%
1.6 Comments and whitespace			100%
1.7 Errors and warnings	100%		100%
1.8 Computers and programs (general)			100%
1.9 Computer tour			100%

**CS 161A: Programming and Problem Solving 1**  
Expires Apr 14th, 2022

**View my activity**  
Select date and time below to show all activity up until a specific time. Default is current time.

Jan 16th, 2022 11:59 pm PST

**Download** a report of my activity.

My activity My subscription Assignments

Assigned zyLabs completion screenshot:

The screenshot displays the zyBooks platform for the course 'CS 161A: Programming and Problem Solving 1'. The main content area features a 'Table of contents' with the following items and their completion status:

Topic	Progress
1. CS 161A: Introduction to C++	100%
1.1 Algorithms	No activities
1.2 Flowcharts	No activities
1.3 Algorithm Example: Calculating birth year	100%
1.4 Programming (general)	100%
1.5 Programming basics	100%
1.6 Comments and whitespace	100%
1.7 Errors and warnings	100%
1.8 Computers and programs (general)	100%
1.9 Computer tour	100%

The sidebar on the right includes the course title 'CS 161A: Programming and Problem Solving 1', the expiration date 'Expires Apr 14th, 2022', and a 'View my activity' section with a date and time selector set to 'Jan 16th, 2022' at '11:59 pm' PST. A 'Download' button is also present for generating a report of activity.

## 2. Program Description

In the box below, describe the purpose of the program. You must include a detailed description with at least two complete sentences.

### Program description:

There are many people that walk and run. While sometime you want to meet the 10,000step recommend to adult. But if the user want to know how many steps they took in miles.

## 3. Sample Run

If you are designing your own program, you will start with a sample run. Imagine a user is running your program - what will they see? What inputs do you expect, and what will be the outputs from the given inputs? Choose test data you will use to test your program. Calculate and show the expected outputs. Use the sample run to test your program.

### Sample run:

Welcome to steps to miles converter

Enter the amount of step taken

You have traveled “ ##mile”

Thank you for using Miles converter

## 4. Algorithmic Design

Before you begin coding, **you must first plan out the logic** and think about what data you will use to test your program for correctness. All programmers plan before coding - this saves a lot of time and frustration! Use the steps below to identify the inputs and outputs, calculations, and steps needed to solve the problem.

### Algorithmic design:

- a. Identify and list all of the user input and their data types.

Step taken- user input

- b. Identify and list all of the user output and their data types.

Welcome to steps to miles converter Display

Enter the amount of step taken Display

You have traveled “.... mile’ Display

Thank you for using Miles converter Display

- c. What calculations do you need to do to transform inputs into outputs? List all formulas needed, if applicable. If there are no calculations needed, state there are no calculations for this algorithm.

Miles = userput/2112

cin<< “Welcome to steps to miles converter”;

Input steps taken

- d. Design the logic of your program using pseudocode or flowcharts. Here is where you would use conditionals, loops or functions (if applicable) and list the steps in transforming inputs into outputs. Walk through your logic steps with the test data from the assignment document or the sample run above.

```
Main(){  
    Int step  
    cout<<"Welcome to steps to miles converter";  
    cin>> step  
    Set steps = steps/2112  
    cout<< "You have traveled:" << steps << "Miles" << endl;  
    return 0  
}
```

## 5. Pseudocode Syntax

Think about each step in your algorithm as an action and use the verbs below:

To do this:	Use this verb:	Example:
Create a variable	DECLARE	DECLARE integer num_dogs
Print to the console window	DISPLAY	DISPLAY "Hello!"
Read input from the user into a variable	INPUT	INPUT num_dogs
Update the contents of a variable	SET	SET num_dogs = num_dogs + 1
<b>Conditionals</b>		

Use a single alternative conditional	IF <i>condition</i> THEN <i>statement</i> <i>statement</i> END IF	IF num_dogs > 10 THEN DISPLAY "That is a lot of dogs!" END IF
Use a dual alternative conditional	IF <i>condition</i> THEN <i>statement</i> <i>statement</i> ELSE <i>statement</i> <i>statement</i> END IF	IF num_dogs > 10 THEN DISPLAY "You have more than 10 dogs!" ELSE DISPLAY "You have ten or fewer dogs!" END IF
Use a switch/case statement	SELECT <i>variable or expression</i> CASE <i>value_1</i> : <i>statement</i> <i>statement</i> CASE <i>value_2</i> : <i>statement</i> <i>statement</i> CASE <i>value_2</i> : <i>statement</i> <i>statement</i> DEFAULT: <i>statement</i> <i>statement</i> END SELECT	SELECT num_dogs CASE 0: DISPLAY "No dogs!" CASE 1: DISPLAY "One dog.." CASE 2: DISPLAY "Two dogs.." CASE 3: DISPLAY "Three dogs.." DEFAULT: DISPLAY "Lots of dogs!" END SELECT
<b>Loops</b>		
Loop while a condition is true - the loop body will execute 0 or more times.	WHILE <i>condition</i> <i>statement</i> <i>statement</i> END WHILE	SET num_dogs = 1 WHILE num_dogs < 10 DISPLAY num_dogs, " dogs!" SET num_dogs = num_dogs + 1 END WHILE
Loop while a condition is true - the loop body will execute 1 or more times.	DO <i>statement</i> <i>statement</i> WHILE <i>condition</i>	SET num_dogs = 1 DO DISPLAY num_dogs, " dogs!" SET num_dogs = num_dogs + 1 WHILE num_dogs < 10
Loop a specific number of times.	FOR <i>counter</i> = <i>start</i> TO <i>end</i> <i>statement</i> <i>statement</i> END FOR	FOR count = 1 TO 10 DISPLAY num_dogs, " dogs!" END FOR
<b>Functions</b>		
Create a function	FUNCTION <i>return_type</i> <i>name (parameters)</i> <i>statement</i> <i>statement</i> END FUNCTION	FUNCTION Integer add(Integer num1, Integer num2) DECLARE Integer sum SET sum = num1 + num2

		RETURN sum END FUNCTION
Call a function	CALL <i>function_name</i>	CALL add(2, 3)
Return data from a function	RETURN <i>value</i>	RETURN 2 + 3