# **83%**

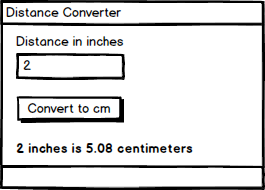
# **Ranken Technical College - C# Chapters 1-5 Hands-On Test Grading Rubric**

This rubric outlines the criteria for grading your C# WinForms GUI-based programs, totaling **100 points**.

## **I. Overall Solution Structure (10 pts)**

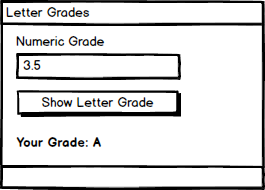
* **5 pts:** All three projects (Distance Converter, GPA Converter, Lawn Mowing Service) are correctly located within a **single Visual Studio solution**.
* **5 pts:** The completed solution is placed in the requested repository location: \HandsOnTest\CH05\.

## **II. Exercise 1: Distance Converter (20 pts)**



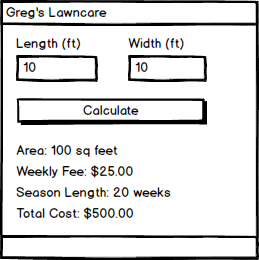
* **5 pts:** Program contains a **constant named CM\_PER\_IN** (e.g., public const double CM\_PER\_IN = 2.54;) with the correct value (2.54 cm per inch).
* **3 pts:** Accepts a numerical distance in inches from the user via a **text input field** (e.g., TextBox).
* **5 pts:** **Correctly converts** the input distance from inches to centimeters using the defined constant.
* **5 pts:** **Displays the output** in the exact desired format: "X inches is Y centimeters" (e.g., "2 inches is 5.08 centimeters").
* **2 pts:** **Input Validation/Error Handling:** Program gracefully handles non-numeric or invalid input (e.g., displays a user-friendly error message, prevents application crash) when the conversion button is pressed.

## **III. Exercise 2: GPA to Letter Grade Converter (30 pts)**



* **3 pts:** Uses a **C# WinForms GUI-based interface** for user interaction.
* **4 pts:** Accepts a numerical GPA (expected range 0.0-4.0) from the user via a **text input field**.
* **6 pts:** **Correctly implements the conditional logic** (e.g., if-else if statements) to determine the letter grade based on the provided conversion table.
* **3 pts:** **Correct Output - 'A':** Produces the correct 'A' grade for grade > 3.0.
* **3 pts:** **Correct Output - 'B':** Produces the correct 'B' grade for 3.0 >= grade > 2.0.
* **3 pts:** **Correct Output - 'C':** Produces the correct 'C' grade for 2.0 >= grade > 1.0.
  + **2.**0 gave a B and 1.0 gave a C
* **3 pts:** **Correct Output - 'D':** Produces the correct 'D' grade for 1.0 >= grade > 0.0.
* **3 pts:** **Correct Output - 'F':** Produces the correct 'F' grade for grade <= 0.0.
* **2 pts:** **Input Validation/Error Handling:** Program gracefully handles non-numeric or out-of-range GPA input (e.g., displays an error message for values less than 0.0 or greater than 4.0). 10 gave an A

## **IV. Exercise 3: Lawn Mowing Service Calculator (30 pts)**



* **5 pts:** Accepts numerical **length and width** of the lawn in feet from the user via separate text input fields.
* **5 pts:** **Area Calculation & Display:** Correctly calculates the total area of the lawn in square feet (length \* width) and displays this area to the user.
* **5 pts:** **Conditional Logic Implementation:** Correctly implements the conditional logic (e.g., if-else if statements) to determine the weekly mowing fee based on the lawn size categories.
* **3 pts:** **Weekly Fee - Large Lot:** Displays the correct weekly fee ($50) for lots that are 600 square feet or more.
* A screenshot of a calculator

  AI-generated content may be incorrect.
* **3 pts:** **Weekly Fee - Medium Lot:** Displays the correct weekly fee ($35) for lots that are 400 square feet or more, but under 600 square feet.
* **3 pts:** **Weekly Fee - Small Lot:** Displays the correct weekly fee ($25) for lots under 400 square feet.
* **4 pts:** **Total Season Fee Calculation & Display:** Correctly calculates the total fee for the 20-week season (based on the determined weekly fee) and displays this total fee to the user.
  + 40x10 should be $750 – program gave $8000
  + 40x10 should be $1000 – program gave $10,000
  + 10x10 should be $500 – program gave $2000
* **2 pts:** **Input Validation/Error Handling:** Program gracefully handles non-numeric or non-positive length/width input (e.g., displays an error message, prevents calculation with invalid dimensions – Negative # calculated dimensions).
* A screenshot of a calculator

  AI-generated content may be incorrect.

## **V. General Code Quality & Best Practices (10 pts)**

* **5 pts:** **Readability:** Code is well-formatted, uses meaningful variable and control names (e.g., txtInchesInput, lblCentimetersOutput), and includes appropriate comments where necessary to explain complex logic.
* **5 pts:** **Usability:** The GUI layout for each program is clear, intuitive, and user-friendly (e.g., clear labels for input fields, logical placement of buttons and output displays).