Teaching Case

Flowcharting Logic: A Language-Independent Approach to Teaching Control Structures in Introductory Programming Courses

Katherine E. Carl
kcarl@wtamu.edu
Computer Information Systems Department
West Texas A&M University
Canyon, Texas 79016, USA

Abstract

The following is a comprehensive set of classroom exercises and assessments that can be used to teach control structures using flow charts in introductory computer programming classes, regardless of language. Each exercise is designed to train students to think through the logic of problems before attempting to write code. Materials are provided via a link to a repository in Appendix A. Video recordings demonstrating the use of the materials are provided via links in Appendix B.

Keywords: flow charts, programming logic, control structures, computer programming, problem-based learning

1. INTRODUCTION

Students in introductory programming courses are tasked with simultaneously learning both the syntax of programming language and the logic of control structures. They are thus often intimidated and overloaded by the programming language syntax and consequently cannot focus on the problem-solving aspects of writing computer applications.

This case presents an approach to teaching introductory programming in which the programming logic and syntax are separated. The logic of programming control structures is addressed first through a focus on creating flow charts rather than writing code to model applications. The first month of the course in which these materials were used was dedicated to the creation of flow charts that applied each of the basic control structures.

The time spent teaching students programming syntax was shortened, as they only needed to be refreshed on the flow chart logic used to represent each control structure they were coding.

The main objectives of this approach are as follows: 1) to help students think through the logic of programs before they write code, 2) to help students that may be intimidated by programming syntax by teaching first without code, and 3) to help students better communicate how their programs function, especially to a non-technical audience.

A comprehensive set of flow chart exercises and solutions is provided for each control structure with instructions for delivery. Supplementary materials, including slides, assignments, and exams, are provided in Appendix A. Video recordings demonstrating the application of the exercises are provided in Appendix B.

2. FLOW CHARTS

Before they can begin creating flow charts from scenarios, students must first be familiar with the notation used in flow charts for programming concepts, and also have access to a tool that can make their flow charts look professional and uniform in style.

Flow Chart Notation

Students were provided with a laminated copy of the reference sheet shown in Figure 1. A larger copy for use is provided in the repository in Appendix A. Students were given permanent markers with which they wrote their names on the laminated cards and were instructed to bring them to class every day for admittance. The card contains all of the symbols and concepts needed to create flow charts for control structures.

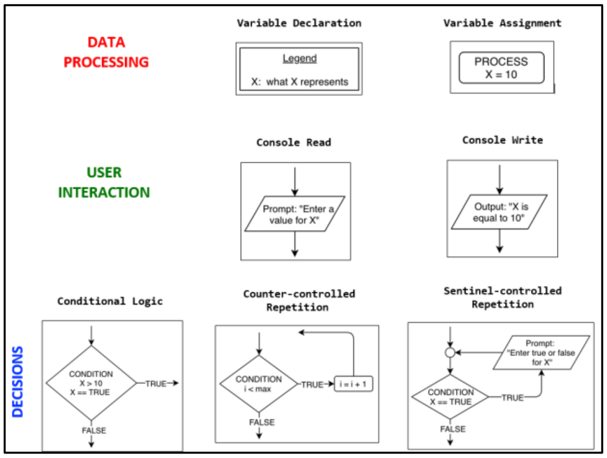

Figure 1: Student Reference Card

Flow Chart Tools

The draw.io tool is free and has both an online and desktop version that can be downloaded at https://www.draw.io/.

Lucid Chart is an online tool that may also be used for free with an education account but does not allow users to have unlimited documents. Download Lucid Chart at https://www.lucidchart.com/pages/.

3. INPUT, PROCESS, AND OUTPUT

Before presenting flow charts to the students, the concepts of 1) variables as programming constructs and 2) computer applications as human-computer interaction facilitated by applications were demonstrated. The top two rows on the students' logic card provide the flow chart analogues for these concepts, as shown in Figure 2.

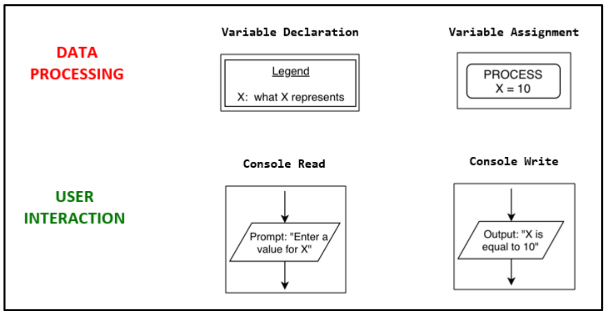

Figure 2: Flow Chart Components for Data Processing and User Interaction

Variables

An object lesson on variables can be delivered using three or four plastic kitchen storage containers of various shapes or sizes. The objective of this lesson is to demonstrate to students that variables may have 1) a name, 2) a type, and 3) a value.

Explain to the students that variables are containers, like the plastic containers. When a variable is declared, is it analogous to a new plastic container being brought into existence. After the container is procured, write a name for it on a small piece of paper and tape it to the side. Explain that this is the name of the variable. Variable declaration maps to the placement of a variable in the flow chart legend, as shown in Figure 2. The purpose of the flow chart legend is to provide the reader with context of the logic presented.

Further explain that the size or shape of the container represents the type of data that the variable can hold. For example, a short circular container may hold only integers, while a short rectangular container may hold variables with fractional parts. The type is not included in the flow chart at this point but may be used when discussing language syntax alongside a flow chart application.

Finally explain that the container is currently empty, but that something can be put into it that matches its type. It can be assigned a value, analogous to calling a process as shown in Figure 2. Write a value on another piece of paper and demonstrate that it can be put into the container. You may replace the current piece of paper with another to demonstrate calling another process of variable reassignment.

User Input and Computer Output

An object lesson on the role and use of input and output in applications can be provided through the in-class assignment of acting or role-playing exercises. This video clip can be used to introduce the idea of human-computer communication: https://www.youtube.com/watch?v=_TNN9xV2MkI. Similar clips from popular culture may be applied depending on the students.

Provide the following prompt to the students:

Let's write a skit. Take about 3 – 5 minutes:

- *Get into pairs and decide who will play the part of the USER and who will play the part of the COMPUTER*
- *Produce a written script in which you capture the following scenario:*

It's [current year] and there is a user that doesn't know how to multiply a number by 2 – they need a computer to do that for them. Illustrate how this user uses the computer to get the desired result.

When most of the students are finished, find a pair wiling to demonstrate their skit to the rest of the class and do so. Use the demonstration as an opportunity to emphasize the need for input and output in addition to processing when creating flow charts for applications. You may also ask the students how the processing takes place, invoking the need for variables.

Flow Chart Exercise
Once the students are comfortable with the concepts presented in the first two rows of the reference card in Figure 1, they may attempt the following in-class exercise:

You put some money in a savings account some time ago and you sort of haven't checked it in a long time…

You want to know how much money you have now in the account, and you don't want to do it by hand.

Create a diagram for the dance you might do with your computer to get what you want to get done.

Flow Chart Solution
Use draw.io to walk students through the creation of the flow chart presented in Figure 3.

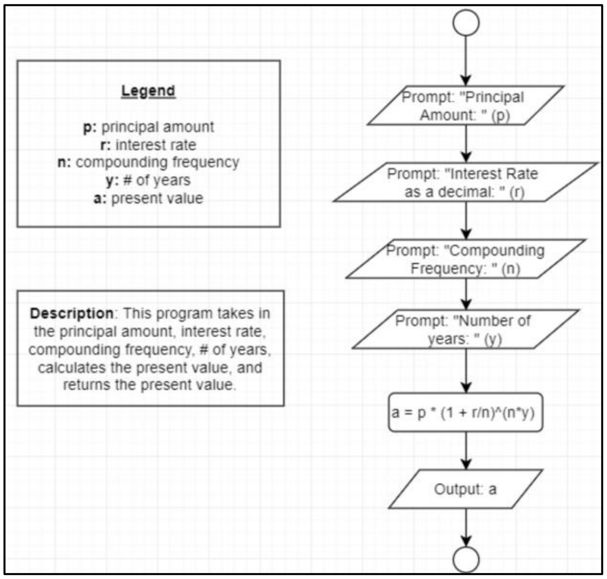
Legend
p: principal amount
r: interest rate
n: compounding frequency
y: # of years
a: present value
Description: This program takes in the principal amount, interest rate, compounding frequency, # of years, calculates the present value, and returns the present value.
Prompt: "Principal Amount: " (p)
Prompt: "Interest Rate as a decimal: " (r)
Prompt: "Compounding Frequency: " (n)
Prompt: "Number of years: " (y)
a = p * (1 + r/n)^(n*y)
Output: a

Figure 3: Input, Process, Output Solution

The assignment that assesses students' understanding of the previous concepts is included in Appendix A.

4. CONDITIONAL LOGIC

Conditional logic without repetition is represented in flow charts as shown in the reference card in Figure 4.

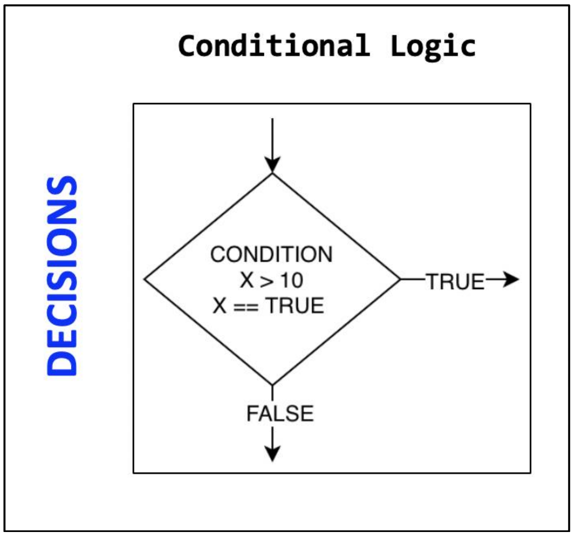

Figure 4: Flow Chart Component for Conditional Logic

Flow Chart Exercise 1
Use the following in-class exercise to teach simple conditional logic:

A donut shop has asked you to design a computer application that determines the price of an order of donuts.

When a customer orders a dozen or more donuts, they get a discount where each donut costs $1.25 instead of the regular $1.50 price.

Flow Chart Solution 1
Use draw.io to walk students through the creation of the flow chart presented in Figure 5.

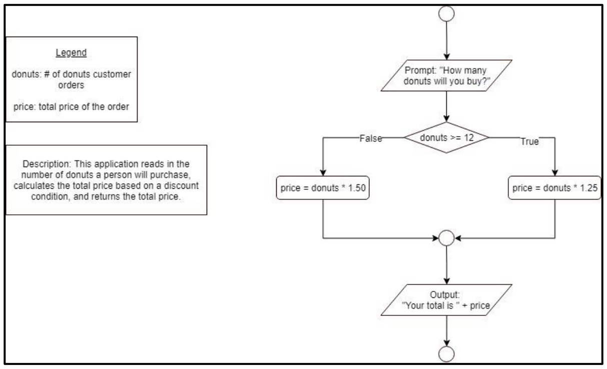
Legend
donuts: # of donuts customer orders
price: total price of the order
Description: This application reads in the number of donuts a person will purchase, calculates the total price based on a discount condition, and returns the total price.
Prompt: "How many donuts will you buy?"
donuts >= 12
False
True
price = donuts * 1.50
price = donuts * 1.25
Output: "Your total is " + price

Figure 5: Basic Conditional Logic Solution

The assignment that assesses students' understanding of the previous concepts is included in Appendix A.

Flow Chart Exercise 2
Use the following in-class exercise to teach conditional logic with cases:

Write an application that asks the user if they would like to eat at the Olive Garden, Texas Roadhouse, or Abuelo's. If the user's response is Olive Garden, tell them to eat there. If not, and the response is Texas Roadhouse, tell them to eat there. If not at Texas Roadhouse, and the response is Abuelo's, tell them to eat there. Then, if none of those three restaurants were entered, tell the user the input was invalid.

Flow Chart Solution 2
Use draw.io to walk students through the creation of the flow chart presented in Figure 6.

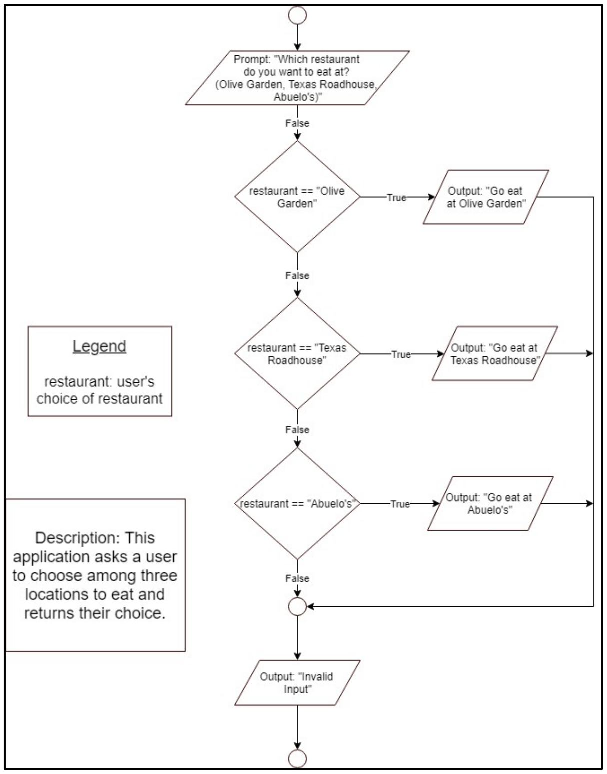
Prompt: "Which restaurant do you want to eat at? (Olive Garden, Texas Roadhouse, Abuelo's)"
False
restaurant == "Olive Garden"
True
Output: "Go eat at Olive Garden"
False
Legend
restaurant: user's choice of restaurant
restaurant == "Texas Roadhouse"
True
Output: "Go eat at Texas Roadhouse"
False
restaurant == "Abuelo's"
True
Output: "Go eat at Abuelo's"
Description: This application asks a user to choose among three locations to eat and returns their choice.
False
Output: "Invalid Input"

Figure 6: Conditional Logic with Cases Solution

The assignment that assesses students' understanding of the previous concepts is included in Appendix A.

5. REPETITION

Once the students are comfortable with representing and identifying the appropriate use of conditional logic without repetition, proceed with the following exercises that reference the flow chart symbols presented in Figure 7 and Figure 8.

Flow Chart Exercise 1
Use the following in-class exercise to teach sentinel-controlled repetition:

Write a "bouncer" application that asks the user how old they are and tells them either that (if they aren't old enough—21) that they need to grow up or (if they are old enough) that they should join the party.

The application should report on invalid input if received.

THE APPLICATION SHOULD RUN AS LONG AS THE USER IS NOT OLD ENOUGH.

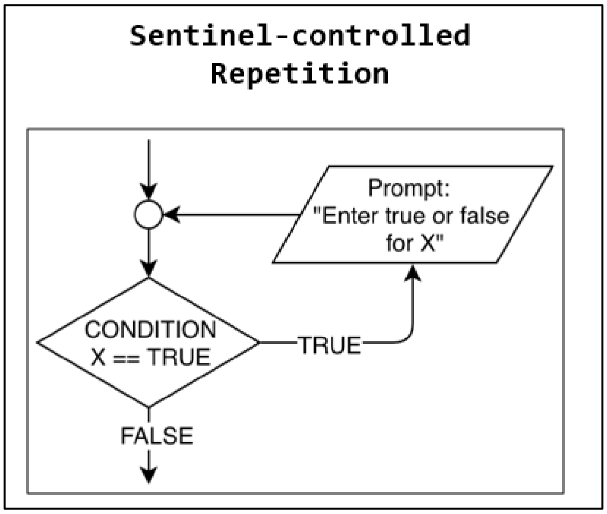

Figure 7: Flow Chart Component for Sentinel-Controlled Repetition

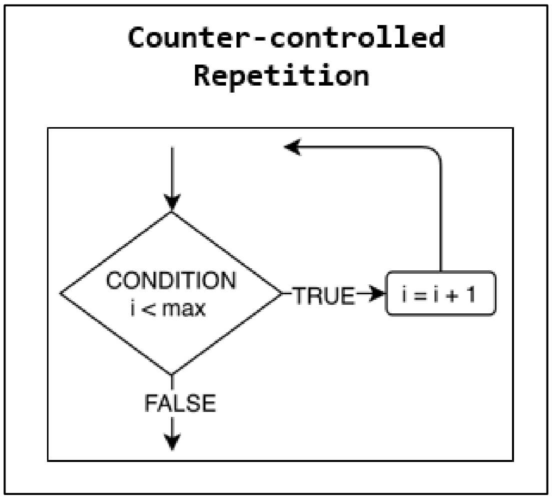

Figure 8: Flow Chart Component for Counter-Controlled Repetition

Flow Chart Exercise 1 Solution
Use draw.io to walk students through the creation of the flow chart presented in Figure 9.

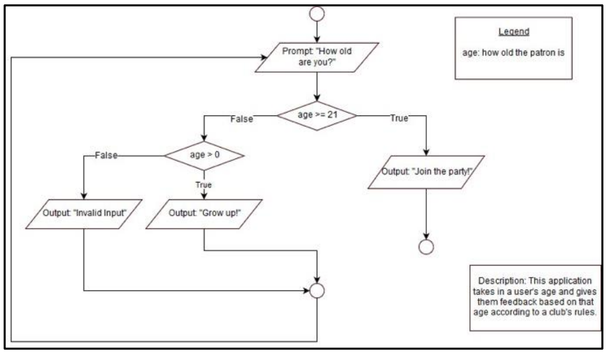
Prompt: "How old
are you?"
Legend
age: how old the patron is
age >= 21
False
True
age > 0
False
True
Output: "Join the party!"
Output: "Invalid Input"
Output: "Grow up!"
Description: This application takes in a user's age and gives them feedback based on that age according to a club's rules.

Figure 9: Sentinel-Controlled Repetition Solution

The assignment that assesses students' understanding of the previous concepts is included in Appendix A.

Flow Chart Exercise 2
Use the following in-class exercise to teach counter-controlled repetition:

Write an application that asks how many times an 8-sided die should be rolled. The die should be rolled that many times and each time it is rolled the number on the face of the die should be returned as output.

When the application is finished running, it should report how many times the number 4 was rolled.

Flow Chart Exercise 2 Solution
Use draw.io to walk students through the creation of the flow chart presented in Figure 10.

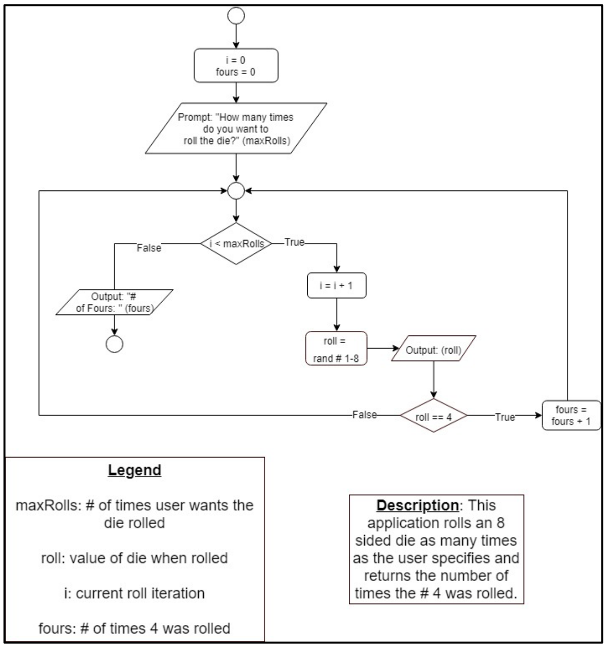
i = 0
fours = 0
Prompt: "How many times
do you want to
roll the die?" (maxRolls)
i < maxRolls
False
True
Output: "#
of Fours: " (fours)
i = i + 1
roll =
rand # 1-8
Output: (roll)
False
roll == 4
True
fours =
fours + 1
Legend
maxRolls: # of times user wants the die rolled
roll: value of die when rolled
i: current roll iteration
fours: # of times 4 was rolled
Description: This application rolls an 8 sided die as many times as the user specifies and returns the number of times the # 4 was rolled.

Figure 10: Counter-Controlled Repetition Solution

The assignment that assesses students' understanding of the previous concepts is included in Appendix A.

6. TRANSITIONING TO CODE

The delivery of the material and the introduction of programming language in an introductory course may be approached in various ways. In this case, the first month of the course was dedicated to flow charts and identifying when each control structure is needed and did not include any programming language. Each flow chart was then revisited and coded by the students as they learned the syntax for each control structure in a particular programming language. However, programming language syntax may also be taught alongside each of the flow chart exercises.

Regardless of delivery, the objective of using flow charts to teach the logic of control structures is to help students become more independent problem-solvers and versatile programmers.

ISSN: 2473-3857
v5 n5114

APPENDIX A: MATERIALS REPOSITORY

Please see the attached repository provided that includes 1) exercises, 2) assignments, 3) exams, and 4) solutions.

https://wtamu0-my.sharepoint.com/:f:/g/personal/kcarl_wtamu_edu/EoLd_Al-fcxItLYGbnV2MYkBxPwkEFaXGjjKEJBdmHqy1A?e=dfQmhR

APPENDIX B: LINKS TO VIDEO DEMONSTRATIONS

Find videos with demonstrations on YouTube Channel playlist here:

https://www.youtube.com/playlist?list=PLYWs0Ua_8loctLYgVPApjKEQuSCAIauhn