pset1: C

Tommy MacWilliam

tmacwilliam@cs50.net

September 11, 2011
Today’s Music

- 3OH!3
  - Deja Vu
  - Hey
  - See you Go
  - Streets of Gold
Axes

- scope: does it have everything?
- correctness: does it work?
- design: how efficiently does it work, and how well is it designed?
- style: how does it look?
Scores

- 5: best
- 4: better
- 3: good
- 2: fair
- 1: poor
Ingredients

- CS50 Appliance: environment
  - https://manual.cs50.net/Appliance
  - having trouble? https://manual.cs50.net/VirtualBox

- gedit: text editor

- Terminal: run programs

- make: source code → machine code
Writing a Program

- use gedit to create hello.c
- make hello
- ./hello
we’ll do it live!
Code style is serious business.
Code Style

Seriously.
CS50 Style Guide

- all your questions answered and more: https://manual.cs50.net/Style
- most important: **be consistent**
  - use the same style decisions everywhere in all programs
  - please :(
Style Examples

▶ example time!
Pennies

jharvard@appliance (~:/pset1): ./pennies
Days in month: 32
Days in month: 31
Pennies on first day: 1
$21474836.47
Input

- `printf`: display message like “Days in month”
- `GetInt()`: prompt the user for an integer
- `int n = GetInt();`
▶ example time!
no input should crash your program!
  - i.e. "This is CS50" days in month and -3.14159 pennies on first day
  - both inputs must be numerical
  - days in month must be valid
    - 28, 29, 30, 31
  - number of pennies must make sense
Validation

- user gave you bad input? loop until input is valid!
- make sure scope of variable is not limited to the loop!

```c
int n = 0;
do {
    n = GetInt();
} while (n is not valid);
```
TODO

- user input (and validation)
- keep track of how much money we have
- double our money the right number of times
- output total
Psuedocode

get number of days in month
get number of pennies on the first day
for (each subsequent day)
    double pennies
    update total
output total dollars and cents
jharvard@appliance (~/pset1): ./greedy
0 hai! How much change is owed? 0.41
4
The Algorithm

- goal: use the fewest coins
- so, make as much change as possible at each step
  - best choice at each step leads to best solution!
The Algorithm

- change for $0.41
  - 1 quarter, 1 coin total, $0.16 left
  - 1 dime, 2 coins total, $0.06 left
  - 1 nickel, 3 coins total, $0.01 left
  - 1 penny, 4 coins total, $0.00 left
Using Loops

▶ try to use each coin until coin is too big
  ▶ use largest coin possible!
Getting Fancy

- %: modulo operator, used calculate remainder
  - \(5 \% 2 = 1\)
  - \(11 \% 3 = 2\)
  - \(3 \% 4 = 3\)

- combine / and % to calculate change
  - division: how many of each coin can be used
  - modulo: how much change is left after coins are used
Floats

- need to convert dollars and cents to just cents
- multiply by 100?
- let’s see...
we need `round()` instead of truncating

- built-in function, just like `printf`
- need to include `<math.h>`

want more info? `man round`
TODO

- user input (and validation)
- keep track of how many coins have been used in total
- keep track of how much change is left to be made
  - try to use each coin, in descending order
- make change until no change is left to be made
- output coins
get money to make change for
convert money to cents
while (more than a quarter left)
  subtract quarter
  increment coins used
while (more than a dime left)
  subtract dime
  increment coins used
...
output coins used
Chart

jharvard@appliance (~/$HOME): ./chart
M spotting F: 3
F spotting M: 4
F spotting F: 1
M spotting M: 2

Who is Spotting Whom
M spotting F
########################
F spotting M
################################
F spotting F
########
M spotting M
###############
Chart

- Chart width is max 80 characters
- Width of bars based on proportion, not total sightings
  - 3 M spotting F is NOT 3 # on the chart
  - Round DOWN when calculating total number of #
Chart Example

- total sightings = 3 + 4 + 1 + 2 = 10
  - M spotting F = 3 / 10 = 30%
  - F spotting M = 4 / 10 = 40%
  - F spotting F = 1 / 10 = 10%
  - M spotting M = 2 / 10 = 20%
max width is 80 characters

- M spotting F = 0.3 \times 80 = 24
- F spotting M = 0.4 \times 80 = 32
- F spotting F = 0.1 \times 80 = 8
- M spotting M = 0.2 \times 80 = 16
TODO

- user input (and validation)
- calculate total sightings
- convert sightings to percentages
- output chart (getting tired of loops yet?)
get M spotting F, F spotting M, etc.
calculate total number of sightings
convert sightings to percentages
convert percentages to number of #s to display
print "M spotting F"
while (# to display for M spotting F)
    print "#"
print "\n"
print "F spotting M"
...