R-Training

PROGRAMS

Programs

- R program - set of instructions for your computer to follow that has been organized into a sequence of steps and cases.
- can create complicated results with the right combination of simple steps
- Strategy for writing a good program:
  - Divide the job into simple tasks
  - Visualizing the relationship between tasks with a flow chart helps.
  - Work on each subtask one at a time.
  - Describe solutions in English, then convert them to R code.
  - Test each solution against concrete examples
  - Once each of the subtasks works, combine the code into a function that can be shared and reused.

The slot machine example

- Types of symbols in slot machine:
  - Diamonds (DD)
  - Sevens (7)
  - Triple bars (BBB)
  - Double bars (BB)
  - Single bars (B)
  - Cherries (C)
  - Zeroes (0)
- A player will win a prize if he gets:
  - Three of the same type of symbol (except for three zeroes)
  - Three bars (of mixed variety)
  - One or more cherries
- Otherwise, the player receives no prize.
- Prize value is determined by the exact combination of symbols
- Value modified by the presence of diamon
  - Wild card: can be considered as any symbol (e.g.: 7 7 DD → 7 7 7) except in case of cher (have to have one real cherry)
  - Higher score: The score is doubled when a set one or more diamonds

Combination	Prize($)
DD DD DD	100
7 7 7	80
BBB BBB BBB	40
BB BB BB	25
B B B	10
C C C	10
Any combination of bars	5
C C *	5
C * C	5
* C C	5
C * *	2
* C *	2
* * C	2

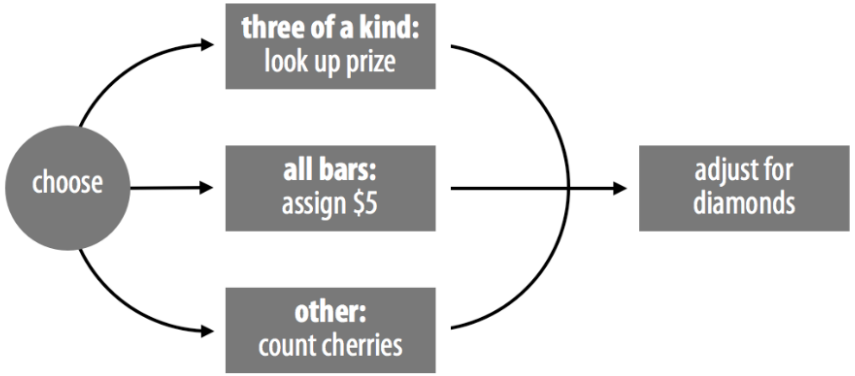
Two types of subtasks:

Sequential steps

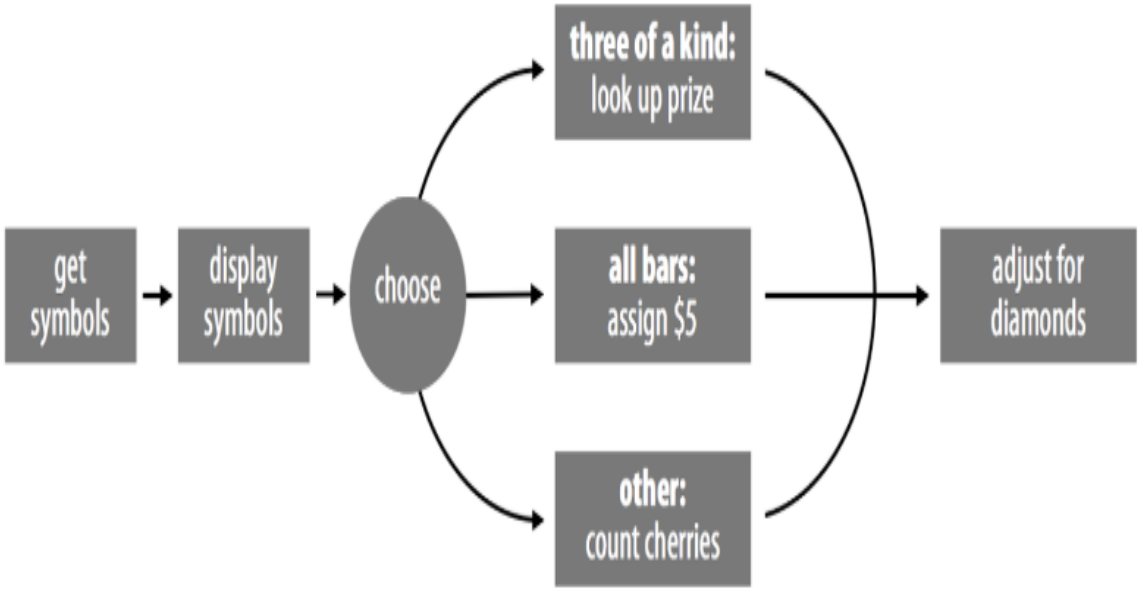
- subdivide a program into a series of sequential steps

Parallel cases

- divide a task to spot groups of similar cases within the task and create code for each task



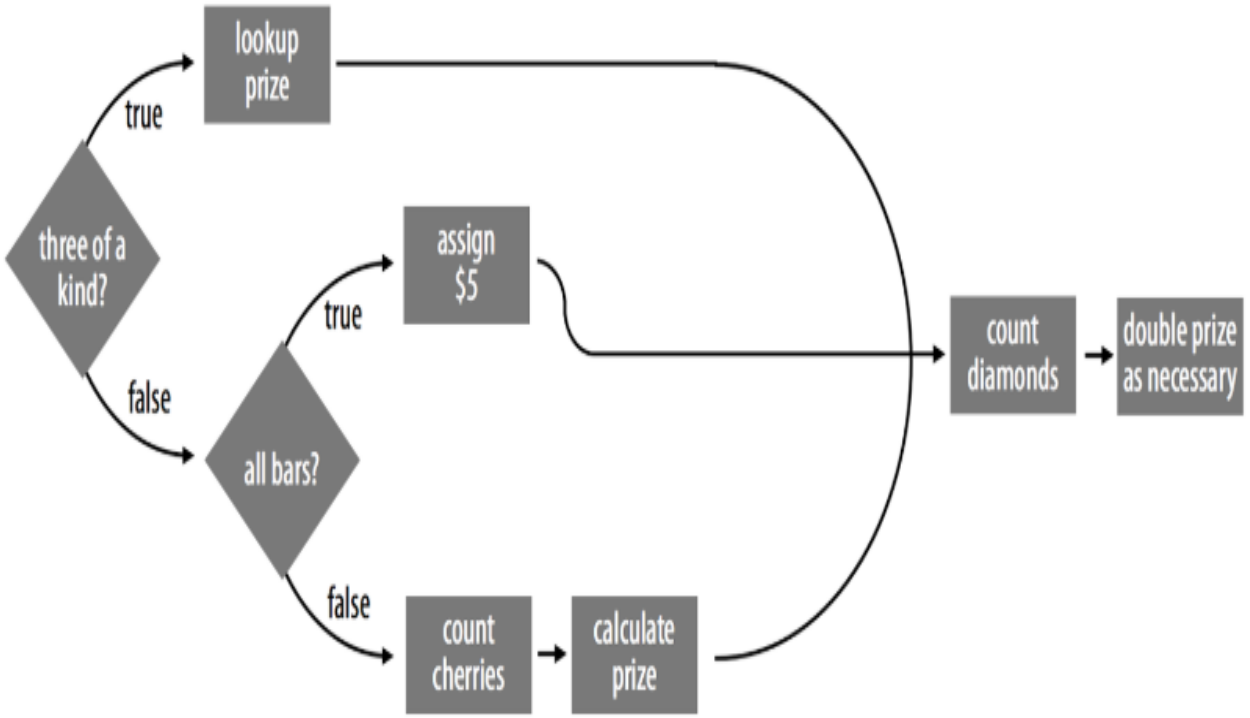
Steps for creating slot machine and scoring



“if” and “else” statements

- Linking cases in parallel requires certain structure where the program has to make choices.
- “if” and “else” statements are logical tests that evaluate a single True or False.
- For “if” statement, if it evaluates to a true, R runs the code given in { } brackets. If the statement evaluates to false, R skips the code and goes to next step. E.g.: If this is true, do plan A.
- “else” statements are used to tell R o run a code if the statement evaluates to a false. E.G.: If this is true, do plan A, else do Plan B
- For more than two mutually exclusive cases, multiple “if” and “else” statements can be used for developing the program.

Flow chart with “if” and “else” statements



1. Test whether the symbols are three of a kind.
2. Test whether the symbols are all bars.
3. Look up the prize for three of a kind based on the common symbol.
4. Assign a prize of $5.
5. Count the number of cherries.
6. Count the number of diamonds.
7. Calculate a prize based on the number of cherries.
8. Adjust the prize for diamonds.

Look up tables

- Sometimes using “if” and “else” statements can lead to length code that is difficult to read and write
- Can use sub-setting to create a vector that captures all the information in the form of look up tables
- Look up tables: R objects that can be used to look up values
- Advantages of look up tables compared to “if” & “else” statements:
  - "if" trees require R to run multiple tests and create unnecessary work
  - “if” trees can be slow to run. Look up tables use R programming strengths to create fast programs.Should not replace every “IF” tree with a look up table
- Should not replace every “IF” tree with a look up table
  - Use “if” tree if each branch of tree runs different code
  - Use look up table if each branch of tree assigns a different value but runs the same code.

Code Comments

- Always use comments to explain the steps in a R program or function
- Makes it easier to understand and breaks long codes to smaller chunks