

Name(s) _____ Period _____ Date _____

Activity Guide - Finding Duplicates



Algorithms and the Problem-Solving Process

The **Problem Solving Process** is useful when planning and writing algorithms. This process will help you clarify and break down a problem into manageable steps so you can easily identify the code you need to write for each step.

Define

- Read the instructions carefully to make sure you understand the goals.
- Rephrase the problem in your own words.
- Identify any new skills you are being asked to apply.
- If there is starter code, read it to understand what it does.

Prepare

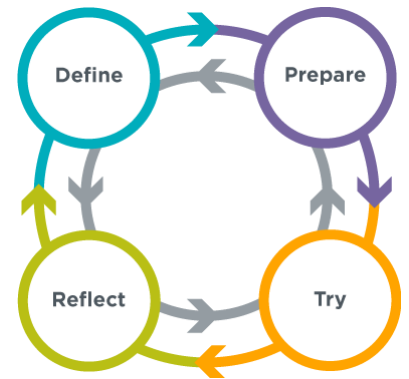
- Write out or draw the steps you need to take to solve the problem.
- List what you already know how to do and what you don't yet.
- Explain your algorithm to a classmate.
- Review similar programs that you've written in the past.

Try

- Write one small piece at a time.
- Test your program often.
- Use comments to document what your code does.
- Go back to a previous step if you get stuck or don't know whether you've solved the problem.

Reflect

- Compare your program to the defined problem to make sure you've solved all aspects of the problem.
- Ask a classmate to try your program and note places where they struggle or show confusion.
- Ask a classmate to read your code to make sure that your documentation is clear and accurate.
- Try to "break" your program to find types of interaction or input that you could handle better.
- Identify changes or improvements you can make next to your program.



Algorithm Planning

For your chosen problem, apply the Problem Solving Process to make sure you understand its requirements, identify similar problems, and plan and refine your algorithm.

Define

Read the instructions carefully to ensure you understand the goals. Rephrase the problem in your own words and identify the **precondition(s)** and **postcondition(s)** of the problem.

Prepare

Write out the steps you need to take to solve the problem.

Questions to Consider:

- Which steps do you already know how to do? Which don't you know how to do yet?
- Explain your steps to a classmate. Was there anything your classmate found confusing? Did they have suggestions that could help improve your algorithm?
- Look at similar programs that you've written in the past. What did you do before that would be helpful in solving this problem?

Try

Write one small piece at a time.

Test your program often!

Use comments to document what your code does.

Go back to the previous steps if you get stuck or aren't sure whether you've solved the problem.

Reflect

Compare your finished program to the defined problem to make sure you've solved all aspects of the problem.

Questions to Consider:

- Ask a classmate to try your program. Are there any parts of your code where they were confused?
- Ask a classmate to read your code to make sure that your documentation is clear and accurate. Do they understand what your code is supposed to do?
- What are some changes or improvements you could make to your program?

What have you accomplished?

What do you need to do next?

Choice A: Forbes Global Companies

The Forbes 2000 Global Companies dataset includes the names of organizations and the year each organization was founded. It has an instance variable for a 1D array of **Company** objects called **companyData**.

The **Company** class represents a company. It has instance variables for the name of the company and the year the company was founded.

Write the **checkForDuplicates()** method to return **true** if any companies in the 1D array **companyData** were founded the same year, otherwise return **false**.

Precondition(s) none

Postcondition(s) companyData is not modified

```
from 0 to companyData.length
  from outer + 1 to companyData.length
    if companyData[outer].getYearFounded() == companyData[inner].getYearFounded()
      return true

return false
```

Choice B: Movies

The TMDb Movie dataset includes the names of movies, the runtime of each movie, and the rating of each movie. It has an instance variable for a 1D array of `Movie` objects called `movieData`.

The `Movie` class represents a movie. It has instance variables for the name of the movie, the length of the movie in minutes, and the rating of the movie.

Write the `checkForDuplicates()` method to return `true` if any movies in the 1D array `movieData` have the same runtime, otherwise return `false`.

Precondition(s) none

Postcondition(s) `movieData` is not modified

```
from 0 to movieData.length
  from outer + 1 to movieData.length
    if movieData[outer].getRuntime() == movieData[inner].getRuntime()
      return true

return false
```

Choice C: Pokemon

The Pokemon Encyclopedia dataset includes the names of Pokemon and the attack points for each Pokemon. It has an instance variable for a 1D array of `Pokemon` objects called `pokemonData`.

The `Pokemon` class represents a Pokemon. It has instance variables for the name of the Pokemon and the attack points the Pokemon has.

Write the `checkForDuplicates()` method to return `true` if any Pokemon in the 1D array `pokemonData` have the same attack points, otherwise return `false`.

Precondition(s) `none`

Postcondition(s) `pokemonData is not modified`

```
from 0 to pokemonData.length
  from outer + 1 to pokemonData.length
    if pokemonData[outer].getAttack() == pokemonData[inner].getAttack()
      return true

return false
```

Choice D: Superheroes

The Super Heroes dataset includes the names of superheroes and the name of the publisher for each superhero. It has an instance variable for a 1D array of **Superhero** objects called **heroData**.

The **Superhero** class represents a superhero. It has instance variables for the name of the superhero and the name of the publisher of the superhero.

Write the **checkForDuplicates()** method to return **true** if any superheroes in the 1D array **heroData** have the same publisher, otherwise return **false**.

Precondition(s) none

Postcondition(s) heroData is not modified

```
from 0 to heroData.length
  from outer + 1 to heroData.length
    if heroData[outer].getPublisher() == heroData[inner].getPublisher()
      return true

return false
```