CPSC 231 Tutorial #20

michael-hung.ca/teaching

Reminders

TODAY

Taffy Tangle Topics

TOMORROW

Assignment 5 Paired Component Due

Overview

— — —

grid
2D Array representation

firstTaffy
keeps track of the indices of the first selected taffy

secondTaffy
keeps track of the indices of the second selected taffy

firstSelected
True if you've already selected a taffy

gameover
True if win/lose condition reached
...

```
...
while not gameover
      was mouse clicked?
            if not firstSelected
                  firstTaffy ← [indices of clicked taffy]
                  firstSelected ← True
            else
                  secondTaffy ← [indices of clicked taffy]
                  if firstTaffy and secondTaffy are adjacent
                        swap them
                        matches exist
                              remove them
                        else
                              swap them back
                  drop taffies and check for new matches
                  firstSelected ← False
      check game win/lose condition
```

Overview

— — —

grid
2D Array representation

firstTaffy
keeps track of the indices of the first selected taffy

secondTaffy
keeps track of the indices of the second selected taffy

firstSelected
True if you've already selected a taffy

gameover
True if win/lose condition reached
...

```
...
while not gameover
      was mouse clicked?
            if not firstSelected
                  firstTaffy ← [indices of clicked taffy]
                  firstSelected ← True
            else
                  secondTaffy ← [indices of clicked taffy]
                  if firstTaffy and secondTaffy are adjacent
                        swap them
                        if matches exist
                              remove them
                        else
                              swap them back
                  drop taffies and check for new matches
                  firstSelected ← False
      check game win/lose condition
```

Overview

— — —

grid
2D Array representation

firstTaffy
keeps track of the indices of the first selected taffy

secondTaffy
keeps track of the indices of the second selected taffy

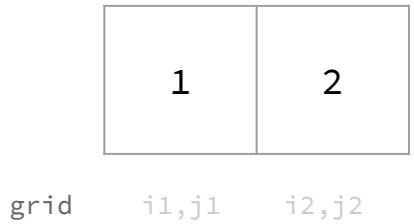
firstSelected
True if you've already selected a taffy

gameover
True if win/lose condition reached
...

```
...
while not gameover
      was mouse clicked?
            if not firstSelected
                  firstTaffy ← [indices of clicked taffy]
                  firstSelected ← True
            else
                  secondTaffy ← [indices of clicked taffy]
                  if firstTaffy and secondTaffy are adjacent
                        swap them
                        if matches exist
                              remove them
                        else
                              swap them back
                  drop taffies and check for new matches
                  firstSelected ← False
      check game win/lose condition
```

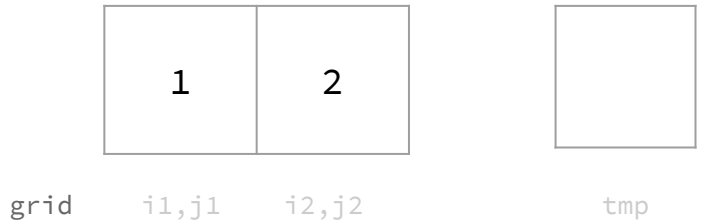
Swapping

```
firstTaffy ← [i1, j1]
secondTaffy ← [i2, j2]
```



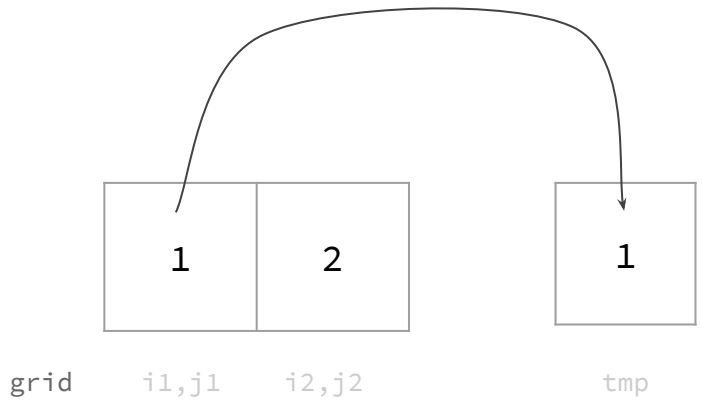
Swapping

```
firstTaffy ← [i1, j1]
secondTaffy ← [i2, j2]

tmp
```



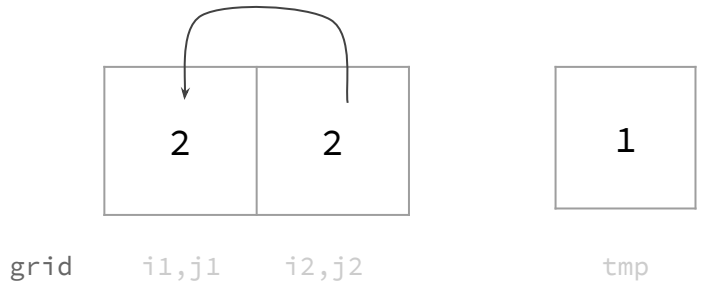
Swapping

```
firstTaffy ← [i1, j1]
secondTaffy ← [i2, j2]

tmp ← grid[i1, j1]
```



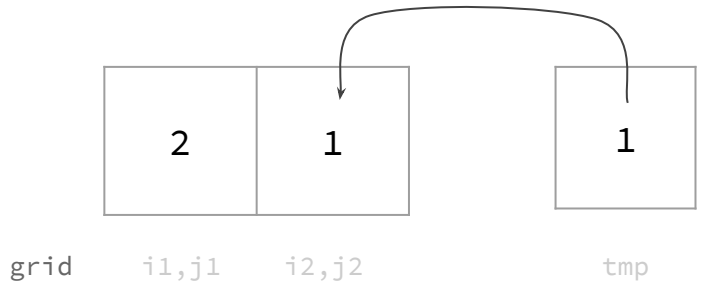
Swapping

```
firstTaffy ← [i1, j1]
secondTaffy ← [i2, j2]

tmp ← grid[i1, j1]

grid[i1, j1] ← grid[i2, j2]
```



Swapping

```
firstTaffy ← [i1, j1]
secondTaffy ← [i2, j2]

tmp ← grid[i1, j1]

grid[i1, j1] ← grid[i2, j2]

grid[i1, j1] ← tmp
```



Overview

— — —

grid
2D Array representation

firstTaffy
keeps track of the indices of the first selected taffy

secondTaffy
keeps track of the indices of the second selected taffy

firstSelected
True if you've already selected a taffy

gameover
True if win/lose condition reached
...

```
...
while not gameover
      was mouse clicked?
            if not firstSelected
                  firstTaffy ← [indices of clicked taffy]
                  firstSelected ← True
            else
                  secondTaffy ← [indices of clicked taffy]
                  if firstTaffy and secondTaffy are adjacent
                        swap them
                        if matches exist
                              remove them
                        else
                              swap them back
                  drop taffies and check for new matches
                  firstSelected ← False
      check game win/lose condition
```

Matching Taffies

Assumption:

- It's impossible to match more than 5 of the same taffy in a line

```
taffiesToRemove ← []
for each taffy in all rows and columns:
     if next four taffies are the same
          taffiesToRemove ∈ their indices
     else if next three taffies are the same
          taffiesToRemove ∈ their indices
     else if next two taffies are the same
          taffiesToRemove ∈ their indices

put 0's in grid at all the indices specified by taffiesToRemove
return len(taffiesToRemove) > 0
```

Overview

— — —

grid
2D Array representation

firstTaffy
keeps track of the indices of the first selected taffy

secondTaffy
keeps track of the indices of the second selected taffy

firstSelected
True if you've already selected a taffy

gameover
True if win/lose condition reached
...

```
...
while not gameover
      was mouse clicked?
            if not firstSelected
                  firstTaffy ← [indices of clicked taffy]
                  firstSelected ← True
            else
                  secondTaffy ← [indices of clicked taffy]
                  if firstTaffy and secondTaffy are adjacent
                        swap them
                        if matches exist
                              remove them
                        else
                              swap them back
                  drop taffies and check for new matches
                  firstSelected ← False
      check game win/lose condition
```

Dropping Taffies

---

while there are 0's in **grid**

- find the *lowest row with a 0*, and **swap** 0's with the space *above* it (y+1)
- replace any 0's in the *highest row in* **grid** with random taffies

Check for matches. If there are any, then repeat all the previous steps.

Reminder: Array Bounds

You need to be constantly checking that you're not trying to access an index of an array that doesn't exist. Use conditionals strategically.

Classes

(Remember my Minesweeper example?)

The Board and Game benefit the most from being a class.

Taffies are simple enough that you don't gain much from making them a class.