File Edit View Help

File Edit View Help

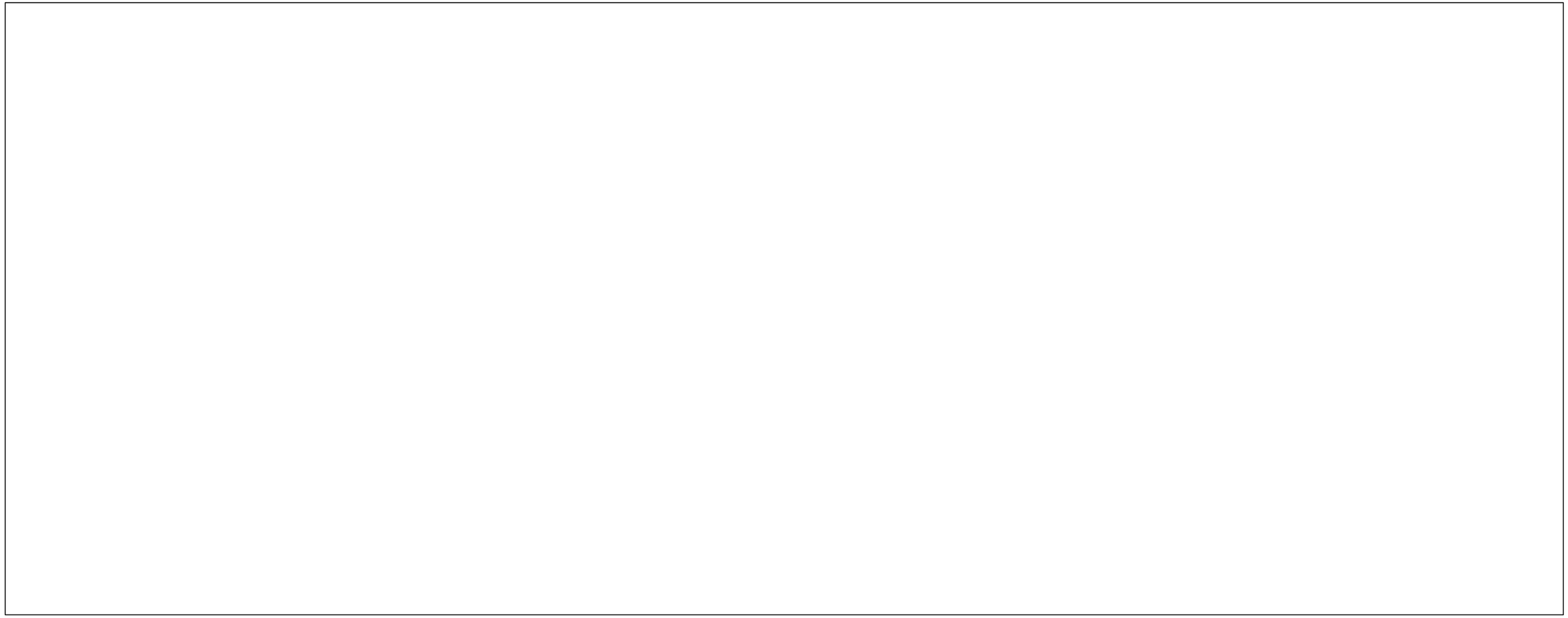
Programming Review
Part 06

File Edit View Help

Introductory Data Structures

File Edit View Help

Problem



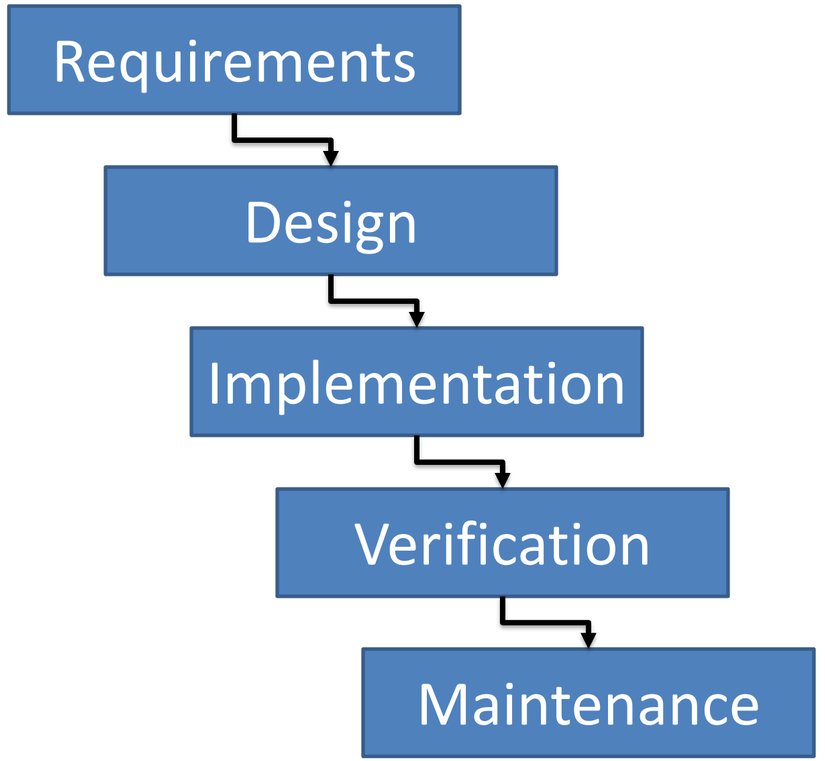
File Edit View Help

Problem

Problem

- Problem to Solve
  - –Keep track of Tacos that I like
- Create Solutions to Problems
- Software Development Models
  - –Agile, Waterfall, Iterative, V-Model, Spiral
- Waterfall Model
  - –Requirements
  - –Design
  - –Implement
  - –Verification
  - –Maintenance

Waterfall Model



File Edit View Help

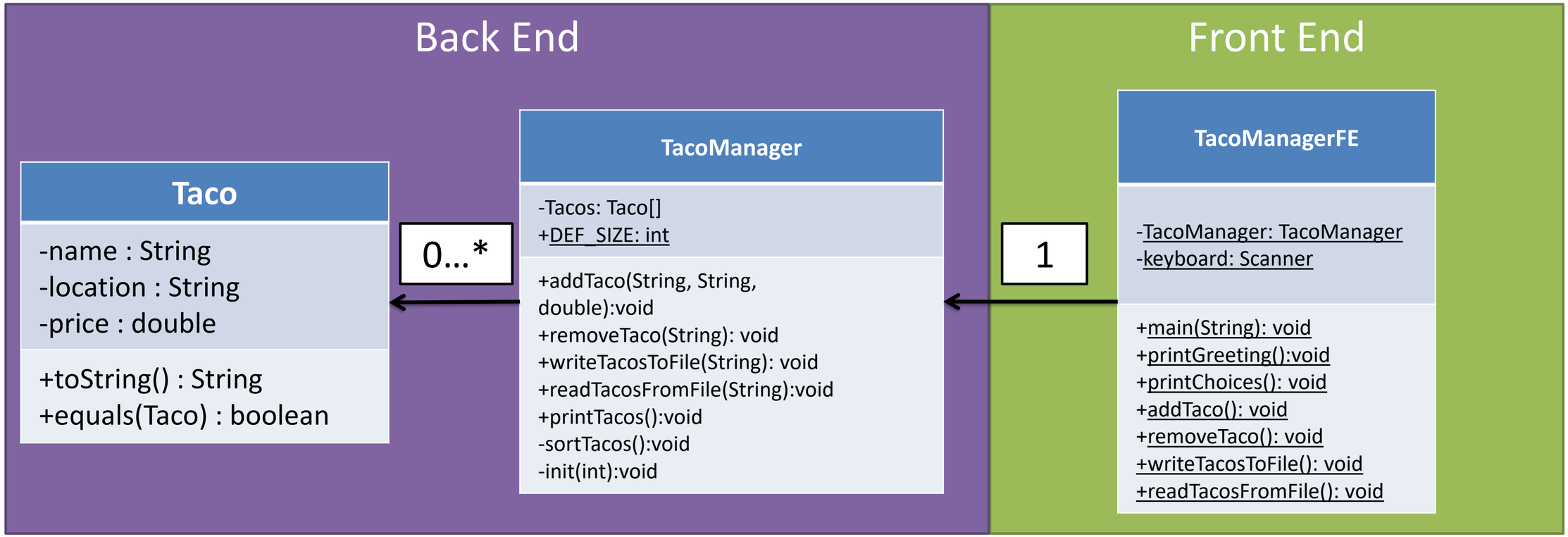
Requirements

- Keep Track of important Taco Information
- Taco's Information
  - Name
  - Location
  - Price

- Should be able to
  - Add a Taco
  - Remove a Taco by its Name
  - Sort all by its Price
  - Write to a Taco File
  - Read from a Taco File
  - Display all Taco information
- Clear and Simple Front End

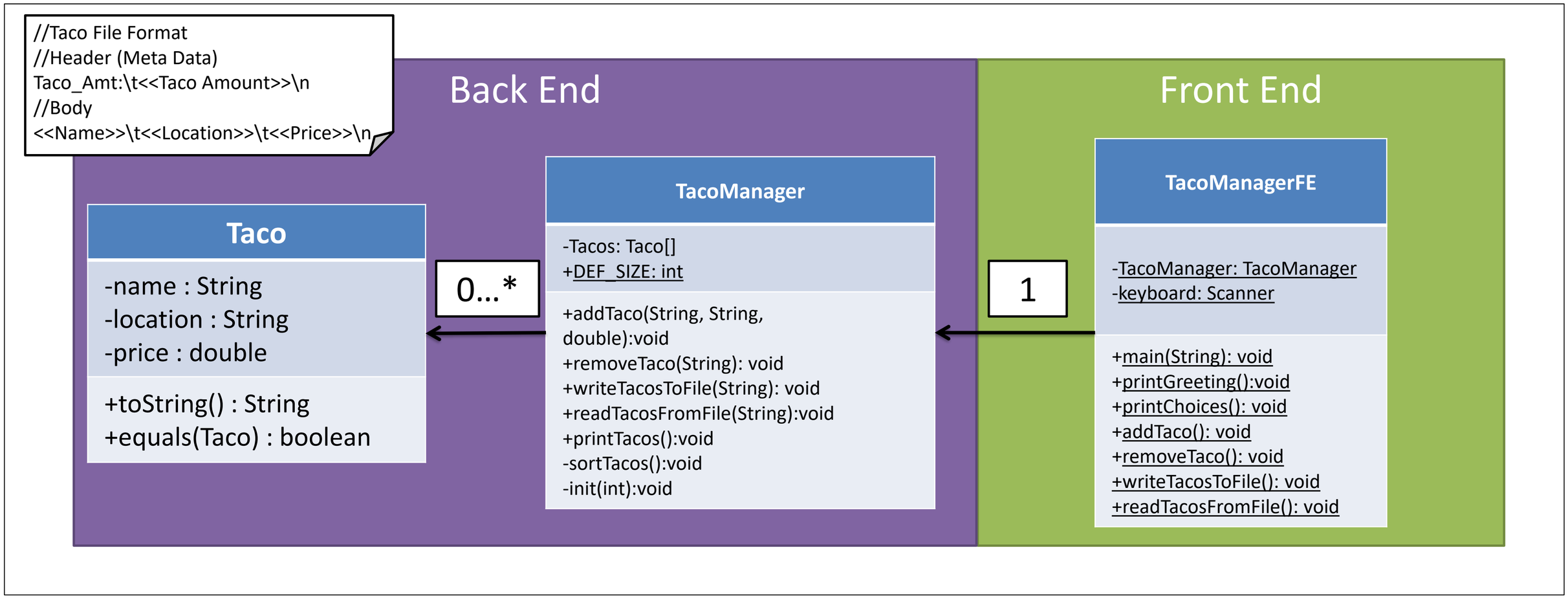
File Edit View Help

Design



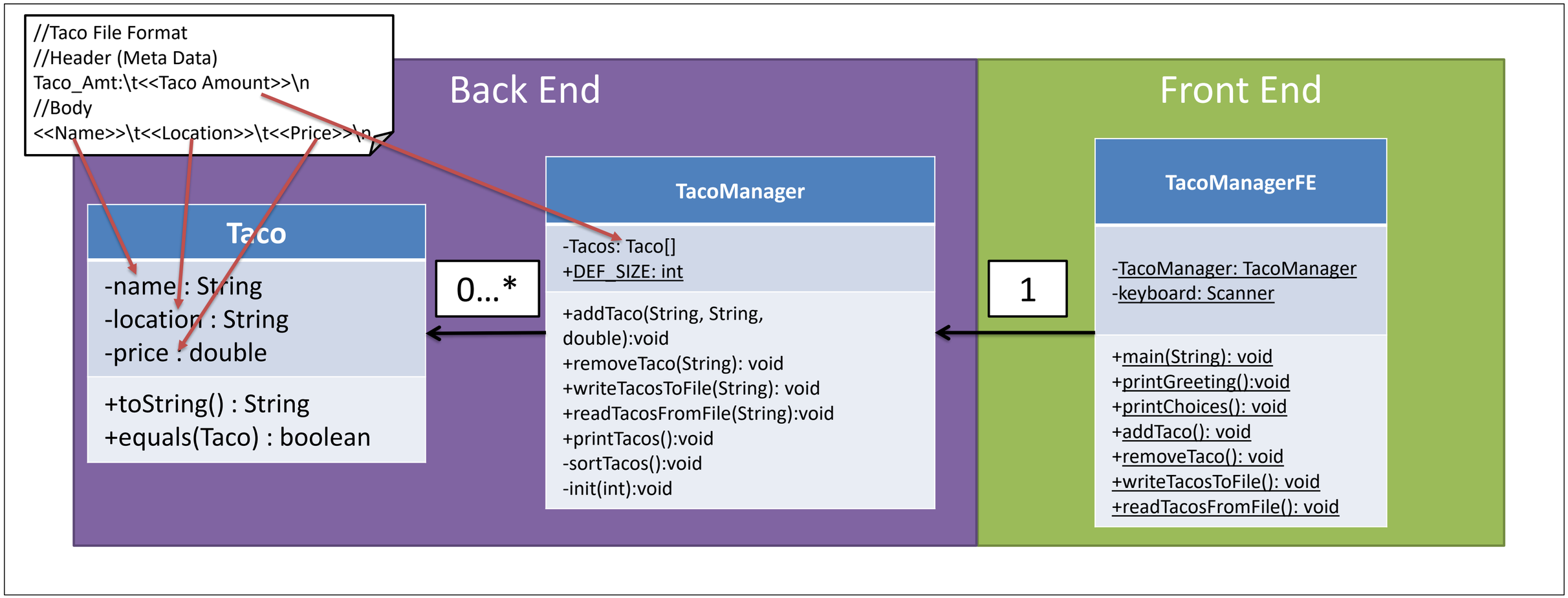
File Edit View Help

Design



File Edit View Help

Design



File Edit View Help

Design

```
Taco_Amt:    3
Taco01       Location01    0.99
Taco02       Location02    3.50
Taco03       Location03    5.00
```

Example File

```
//Taco File Format
//Header (Meta Data)
Taco_Amt:\t<<Taco Amount>>\n
//Body
<<Name>>\t<<Location>>\t<<Price>>\n
```

Back End

Taco

-name : String
-location : String
-price : double

+toString() : String
+equals(Taco) : boolean

0...*

TacoManager

-Tacos: Taco[]
+DEF_SIZE: int

+addTaco(String, String, double):void
+removeTaco(String): void
+writeTacosToFile(String): void
+readTacosFromFile(String):void
+printTacos():void
-sortTacos():void
-init(int):void

Front End

1

TacoManagerFE

-TacoManager: TacoManager
-keyboard: Scanner

+main(String): void
+printGreeting():void
+printChoices(): void
+addTaco(): void
+removeTaco(): void
+writeTacosToFile(): void
+readTacosFromFile(): void

Object Array

- Arrays of Objects are Arrays of **Memory Addresses**
- An Array is a Class Datatype in Java
- The Array's identifier points to the contents of the array
- The Array's indices point to the contents of the constructed and unconstructed Objects
- Default values for Object Arrays are NULL
  - Meaning "Nothing" or "Empty"

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	NULL	40
Tacos[1]	NULL	48
Tacos[2]	NULL	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

File Edit View Help

Initializing the Object Array

Taco Array Structure

- Keep all constructed Objects to one side
  - –Toward the top
- No NULL elements in between constructed Objects
- First NULL Element means everything after that is also assumed NULL
  - –Everything below will also be NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	NULL	40
Tacos[1]	NULL	48
Tacos[2]	NULL	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

File Edit View Help

Add Method Concept

Taco Array Structure

- Adding
  - –Start from the first Index
  - –Find first NULL element
  - –Construct / Assign value to that location

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	NULL	40
Tacos[1]	NULL	48
Tacos[2]	NULL	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

File Edit View Help

Taco Array Structure

- Adding
  - –Start from the first Index
  - –Find first NULL element
  - –Construct / Assign value to that location

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	NULL	40
Tacos[1]	NULL	48
Tacos[2]	NULL	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

Taco Array Structure

- Adding
  - –Start from the first Index
  - –Find first NULL element
  - –Construct / Assign value to that location

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	NULL	40
Tacos[1]	NULL	48
Tacos[2]	NULL	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

File Edit View Help

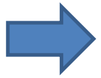
Taco Array Structure

- Adding
  - –Start from the first Index
  - –Find first NULL element
  - –Construct / Assign value to that location

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	256	40
Tacos[1]	NULL	48
Tacos[2]	NULL	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

Taco Array Structure

- Adding
  - –Start from the first Index
  - –Find first NULL element
  - –Construct / Assign value to that location



Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	256	40
Tacos[1]	NULL	48
Tacos[2]	NULL	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

Taco Array Structure

- Adding
  - –Start from the first Index
  - –Find first NULL element
  - –Construct / Assign value to that location

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	256	40
Tacos[1]	NULL	48
Tacos[2]	NULL	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

Taco Array Structure

- Adding
  - –Start from the first Index
  - –Find first NULL element
  - –Construct / Assign value to that location

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	256	40
Tacos[1]	512	48
Tacos[2]	NULL	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

File Edit View Help

Taco Array Structure

- Adding
  - –Start from the first Index
  - –Find first NULL element
  - –Construct / Assign value to that location

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	256	40
Tacos[1]	512	48
Tacos[2]	324	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

File Edit View Help

Add Method Implementation

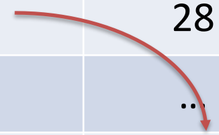
File Edit View Help

Remove Method Concept

Taco Array Structure

- Removing
  - –Start from the first Index
  - –Find the element to remove's index
  - –If not found then return
  - –Then shift over by one (Tacos[i] = Tacos[i+1]
  - –Set last element to NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	256	40
Tacos[1]	512	48
Tacos[2]	324	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112



Taco Array Structure

- Removing
  - –Start from the first Index
  - –Find the element to remove's index
  - –If not found then return
  - –Then shift over by one (Tacos[i] = Tacos[i+1]
  - –Set last element to NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	256	40
Tacos[1]	512	48
Tacos[2]	324	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

Taco Array Structure

- Removing
  - –Start from the first Index
  - –Find the element to remove's index
  - –If not found then return
  - –Then shift over by one (Tacos[i] = Tacos[i+1]
  - –Set last element to NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	NULL	40
Tacos[1]	512	48
Tacos[2]	324	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

Taco Array Structure

- Removing
  - –Start from the first Index
  - –Find the element to remove's index
  - –If not found then return
  - –Then shift over by one (Tacos[i] = Tacos[i+1]
  - –Set last element to NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	NULL	40
Tacos[1]	512	48
Tacos[2]	324	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

Taco Array Structure

- Removing
  - –Start from the first Index
  - –Find the element to remove's index
  - –If not found then return
  - –Then shift over by one (Tacos[i] = Tacos[i+1]
  - –Set last element to NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	256	40
Tacos[1]	512	48
Tacos[2]	324	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

Taco Array Structure

- Removing
  - –Start from the first Index
  - –Find the element to remove's index
  - –If not found then return
  - –Then shift over by one (Tacos[i] = Tacos[i+1]
  - –Set last element to NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	256	40
Tacos[1]	512	48
Tacos[2]	324	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

Taco Array Structure

- Removing
  - –Start from the first Index
  - –Find the element to remove's index
  - –If not found then return
  - –Then shift over by one (Tacos[i] = Tacos[i+1]
  - –Set last element to NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	512	40
Tacos[1]	324	48
Tacos[2]	NULL	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

Taco Array Structure

- Removing
  - –Start from the first Index
  - –Find the element to remove's index
  - –If not found then return
  - –Then shift over by one (Tacos[i] = Tacos[i+1]
  - –Set last element to NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	324	40
Tacos[1]	NULL	48
Tacos[2]	NULL	54
…	…	…
Tacos[8]	NULL	104
Tacos[9]	NULL	112

Taco Array Structure

- Removing
  - –Start from the first Index
  - –Find the element to remove's index
  - –If not found then return
  - –Then shift over by one (Tacos[i] = Tacos[i+1]
  - –Set last element to NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	256	40
Tacos[1]	512	48
Tacos[2]	324	54
…	…	…
Tacos[8]	4096	104
Tacos[9]	8192	112

Taco Array Structure

- Removing
  - –Start from the first Index
  - –Find the element to remove's index
  - –If not found then return
  - –Then shift over by one (Tacos[i] = Tacos[i+1]
  - –Set last element to NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	512	40
Tacos[1]	324	48
Tacos[2]	1729	54
…	…	…
Tacos[8]	8192	104
Tacos[9]	8192	112

Taco Array Structure

- Removing
  - –Start from the first Index
  - –Find the element to remove's index
  - –If not found then return
  - –Then shift over by one (Tacos[i] = Tacos[i+1]
  - –Set last element to NULL

Identifier	Contents	Byte Address
…	…	…
Tacos	40	28
…	…	…
Tacos[0]	512	40
Tacos[1]	324	48
Tacos[2]	1729	54
…	…	…
Tacos[8]	8192	104
Tacos[9]	NULL	112

File Edit View Help

Remove Method Implementation

File Edit View Help

File I/O Concept

File Edit View Help

Design

```
Taco_Amt:    3
Taco01       Location01    0.99
Taco02       Location02    3.50
Taco03       Location03    5.00
```

Example File

```
//Taco File Format
//Header (Meta Data)
Taco_Amt:\t<<Taco Amount>>\n
//Body
<<Name>>\t<<Location>>\t<<Price>>\n
```

Back End

Taco

-name : String
-location : String
-price : double

+toString() : String
+equals(Taco) : boolean

0...*

TacoManager

-Tacos: Taco[]
+DEF_SIZE: int

+addTaco(String, String, double):void
+removeTaco(String): void
+writeTacosToFile(String): void
+readTacosFromFile(String):void
+printTacos():void
-sortTacos():void
-init(int):void

Front End

1

TacoManagerFE

-TacoManager: TacoManager
-keyboard: Scanner

+main(String): void
+printGreeting():void
+printChoices(): void
+addTaco(): void
+removeTaco(): void
+writeTacosToFile(): void
+readTacosFromFile(): void

File Edit View Help

TAB '\t' Delimited Example Implementation

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

File

```
Taco_Amt:       3
Taco01  Location01      0.99
Taco02  Location02      3.50
Taco03  Location03      5.00
```

Data

```
nextLine = NULL
splitLines = NULL
name = NULL
location = NULL
price = 0.0
```

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

File

Taco_Amt:	3	
Taco01	Location01	0.99
Taco02	Location02	3.50
Taco03	Location03	5.00

Data

```
nextLine = “Taco\tLocation01\t0.99”
splitLines = NULL
name = NULL
location = NULL
price = 0.0
```

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

File

Taco_Amt:	3	
Taco01	Location01	0.99
Taco02	Location02	3.50
Taco03	Location03	5.00

Data

nextLine = “Taco01\tLocation01\t0.99”

splitLines =

Index	0	1	2
Data	“Taco01”	“Location01”	“0.99”

name = NULL
location = NULL
price = 0.0

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

File

Taco_Amt:	3	
Taco01	Location01	0.99
Taco02	Location02	3.50
Taco03	Location03	5.00

Data

nextLine = “Taco01\tLocation01\t0.99”

splitLines =

Index	0	1	2
Data	“Taco01”	“Location01”	“0.99”

name = NULL
location = NULL
price = 0.0

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

File

Taco_Amt:	3	
Taco01	Location01	0.99
Taco02	Location02	3.50
Taco03	Location03	5.00

Data

nextLine = “Taco01\tLocation01\t0.99”

splitLines =

Index	0	1	2
Data	“Taco01”	“Location01”	“0.99”

name = “Taco01”

location = NULL

price = 0.0

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

File

Taco_Amt:	3	
Taco01	Location01	0.99
Taco02	Location02	3.50
Taco03	Location03	5.00

Data

nextLine = “Taco01\tLocation01\t0.99”

splitLines =

Index	0	1	2
Data	“Taco01”	“Location01”	“0.99”

name = “Taco01”

location = “Location01”

price = 0.0

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

File

Taco_Amt:	3	
Taco01	Location01	0.99
Taco02	Location02	3.50
Taco03	Location03	5.00

Data

nextLine = “Taco01\tLocation01\t0.99”

splitLines =

Index	0	1	2
Data	“Taco01”	“Location01”	“0.99”

name = “Taco01”

location = “Location01”

price = 0.99

Double.parseDouble(“0.99”)

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

Taco
-name = “Taco01” -location = “Location01” -price = 0.99

File

Taco_Amt: 3

Taco01	Location01	0.99
Taco02	Location02	3.50
Taco03	Location03	5.00

Data

nextLine = “Taco01\tLocation01\t0.99”

splitLines =

Index	0	1	2
Data	“Taco01”	“Location01”	“0.99”

name = “Taco01”
location = “Location01”
price = 0.99

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

Taco
-name = “Taco01”
-location = “Location01”
-price = 0.99

File

Taco_Amt: 3

Taco01	Location01	0.99
Taco02	Location02	3.50
Taco03	Location03	5.00

Data

nextLine = “Taco01\tLocation01\t0.99”

splitLines =

Index	0	1	2
Data	“Taco01”	“Location01”	“0.99”

name = “Taco01”

location = “Location01”

price = 0.99

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

File

Taco_Amt:	3	
Taco01	Location01	0.99
Taco02	Location02	3.50
Taco03	Location03	5.00

Data

```
nextLine = NULL
splitLines = NULL

name = NULL
location = NULL
price = 0.0
```

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

File

```
Taco_Amt:       3
Taco01   Location01      0.99
Taco02   Location02      3.50
Taco03   Location03      5.00
```

Data

```
nextLine = NULL
splitLines = NULL

name = NULL
location = NULL
price = 0.0
```

File Edit View Help

TAB Delimited Example Detailed

```
…
while(fileScanner.hasNextLine())
{
    nextLine = fileScanner.nextLine();
    splitLines = nextLine.split(DELIM);
    if(splitLines.length == BODY_AMT)
    {
        String name = splitLines[0];
        String loc = splitLines[1];
        double price = Double.parseDouble(splitLines[2]);
        Taco addTaco = new Taco(name,loc,price);
        this.addTaco(addTaco);
    }
    …
}
```

Taco
-name = “Taco02” -location = “Location02” -price = 3.5

File

Taco_Amt: 3

Taco01	Location01	0.99
Taco02	Location02	3.50
Taco03	Location03	5.00

Data

nextLine = “Taco02\tLocation02\t3.50”

splitLines =

Index	0	1	2
Data	“Taco02”	“Location02”	“3.50”

name = “Taco02”
location = “Location02”
price = 3.5

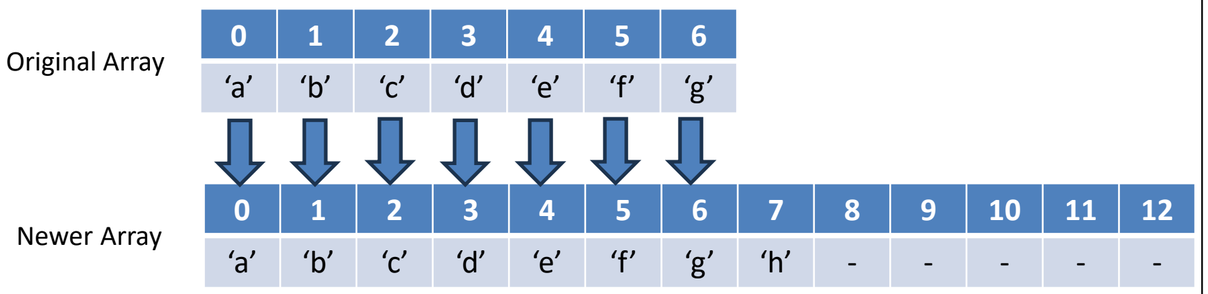
File Edit View Help

Is this the best Implementation?

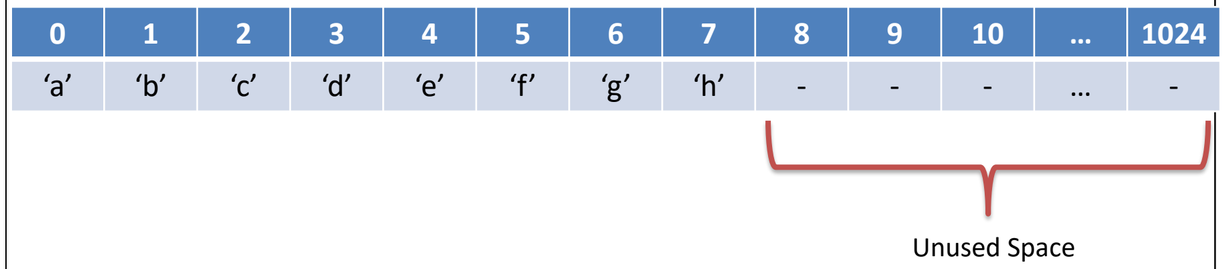
Problems with the Array

- Did we use the correct data structure?
- Arrays Pros
  - –Random Access
  - –Better Data Locality
- Array's Cons
  - –Not resizable
- If we do not know the exact size, then we may
  - –Guess too small => Need create a newer larger array and transfer the data.
  - –Guess too big => Lots of unused, wasted space (Internal Fragmentation).

Array Too Small



Array Too Big



File Edit View Help

Programming Review
Part 06