UML Class Diagrams

Joseph Maples CS101
Design for an object-oriented people database

+	public
-	private
	package
#	protected

legend

+ Driver

+ main(args:String [])
+ sortByName(array: Person[])
+ sortByDate(array: Person[])
+ sortBySalary(array: Person[])
+ sortByAddress(array: Person[])
+ employeeIndicies(array: Person[])

+ Date
- day:int - month:int - year:int
+ Date()

+	getDay():int
-	setDay(day:int)
+	getMonth():int
-	setMonth(month:int)
+	getYear():int
-	setYear(year:int)
+	compareTo(date:Date)
+	toString()

+	Person

#	name:String
#	address:String
#	phoneNumber:String
#	emailAddress:String
#	date:Date

+	Person(String, String, String, String, Date)
#	setName(String)
#	setAddress(String)
#	setEmailAddress(String)
#	setDate(Date)
#	setPhoneNumber(String)
+	getName():String
+	getAddress():String
+	getPhoneNumber():String

```
+   getEmailAddress():String
+   getDate():Date
+   toString()
```

```
+   Employee
```

```
-   office:String
-   salary:double
-   title:String
```

```
+   Employee(String,String,String,String,String,double,String,Date)
+   setOffice(String)
+   setSalary(double)
+   setTitle(String)
+   setYear(String)
+   getOffice():String
+   getSalary():double
+   getTitle():String
```

```
+   Staff
```

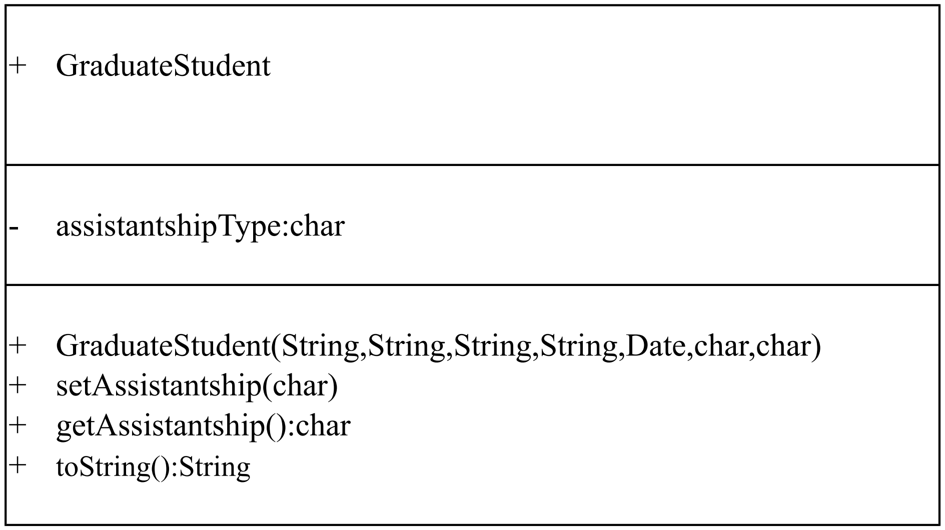
```
-   supervisor:String
```

+	Staff(String,String,String,String,String,double,String,Date,String
+	setSupervisor(String)
+	getSupervisor():String
+	toString():String

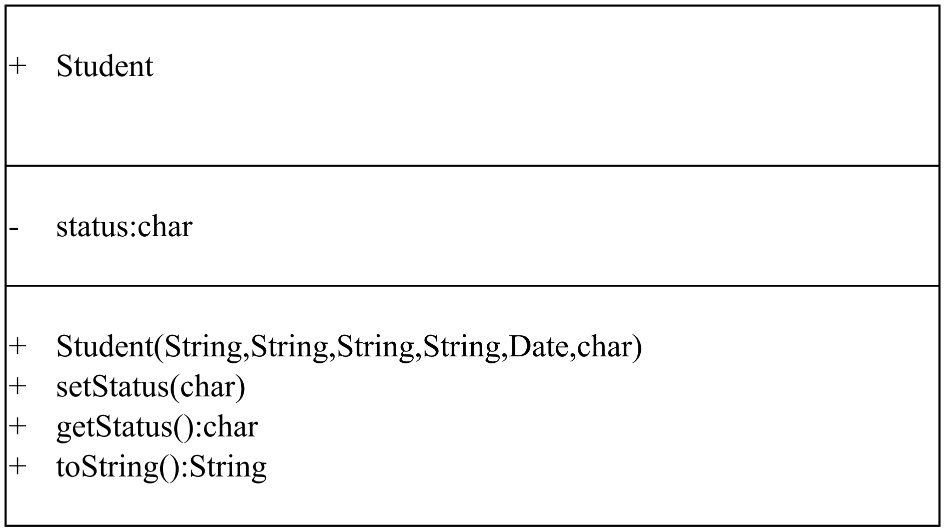
+	Faculty
-	officeHours:String
+	Faculty(String,String,String,String,String,double,String,Date,String
+	setOfficeHours(String)
+	getOfficeHours():String
+	toString():String

+	UndergraduateStudent
+	UndergraduateStudent(String,String,String,String,Date,char)
+	toString():String

UML Class Diagrams

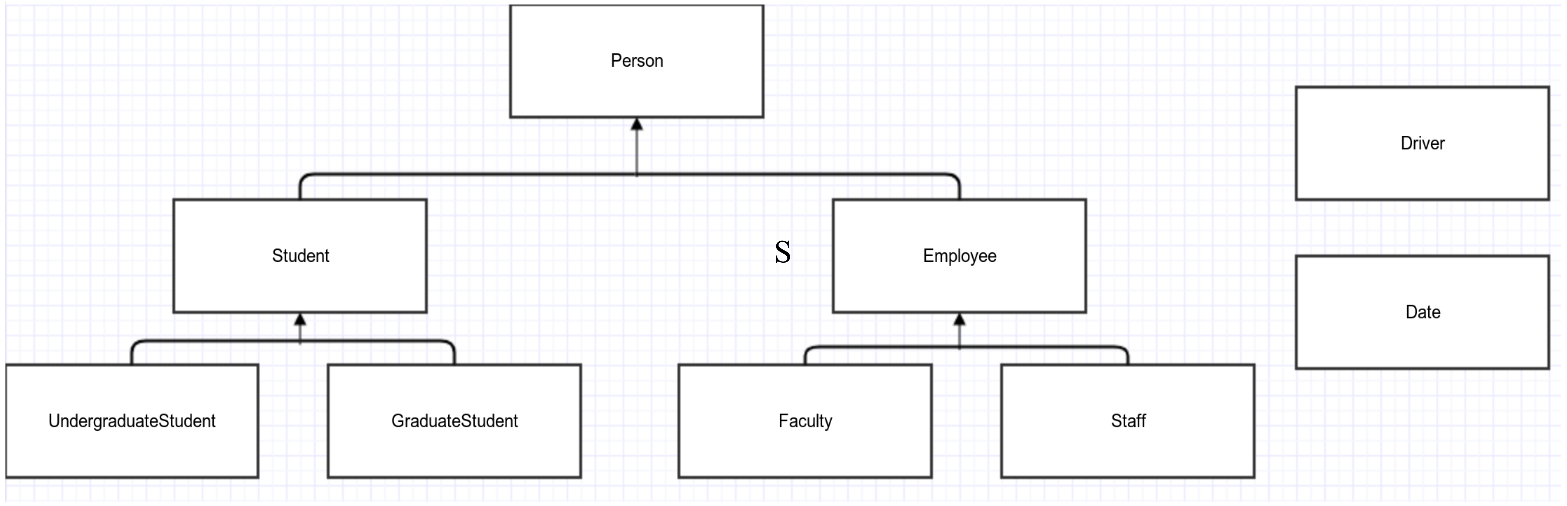


+ GraduateStudent
- assistantshipType:char
+ GraduateStudent(String,String,String,String,Date,char,char)
+ setAssistantship(char)
+ getAssistantship():char
+ toString():String



UML Class Interconnection Diagram

Joseph Maples CS101
Design for an object-oriented people database



Joseph Maples CS101
Design for an object-oriented people database

Data Table for Class Date

Variable or Constant	Type	Purpose
day	int	The day
month	int	The month
year	int	The year

Data Table for toString()

Variable or Constant	Type	Purpose
date	String	The date object as a string

Data Table for Driver

Joseph Maples CS101
Design for an object-oriented people database

Data table for main(args:String [])

Variable or Constant	Type	Purpose
inFile	File	input file
outFile	file	output file
fileScan	Scanner	Scan the input file
printer	PrintStream	Write to the output file
getLines	Scanner	Scanner to get the number of lines in inFile
lines	int	Number of lines in inFile
people	Person[]	The array of persons
index	int	The index of people to set
line	String	The current line of input
data	String[]	The line of input as a word array

Data Table for sortByName(array: Person[])

Variable or Constant	Type	Purpose
array	Person[]	The array of persons
size	int	Size of the array
index	int	The index of people to check
minIndex	int	index of the lowest element
index2	int	The index of people to check in inner for loop
temp	Person	Person to swap

Data Table for sortByDate(array: Person[])

Variable or Constant	Type	Purpose
array	Person[]	The array of persons
size	int	Size of the array
index	int	The index of people to check
minIndex	int	index of the lowest element
index2	int	The index of people to check in inner for loop
temp	Person	Person to swap

Data Table for sortBySalary(array: Person[])

Data Table for Driver

Variable or Constant	Type	Purpose
array	Person[]	The array of persons
employees	int[]	Array containing the indexes of employees
index	int	The index of people to check
minIndex	int	index of the lowest element
index2	int	The index of people to check in inner for loop
temp	Person	Person to swap

Data Table for sortByAddress(array: Person[])

Variable or Constant	Type	Purpose
array	Person[]	The array of persons
size	int	Size of the array
index	int	The index of people to check
minIndex	int	index of the lowest element
index2	int	The index of people to check in inner for loop
temp	Person	Person to swap

Data Table for employeeIndicies(array: Person[])

Variable or Constant	Type	Purpose
array	Person[]	The array of persons
indexString	String	String that contains all indicies that are employees
index	int	The index of people to check
indicies	String[]	Array that contains all indicies that are employees
element	int	The index of people to check
employees	int[]	Array that contains all indicies that are employees

Joseph Maples CS101
Design for an object-oriented people database

Data Table for Class Person

Variable or Constant	Type	Purpose
name	String	Persons name
address	String	Persons address
phoneNumber	String	Persons number
email	String	Persons email
date	Date	A critical date

Data Table for Person(name:String, address:String, phoneNumber:String, email:String, date:Date)

Variable or Constant	Type	Purpose
name	String	Persons name
address	String	Persons address
phoneNumber	String	Persons number
email	String	Persons email
date	Date	A critical date

Data Table for setName()

Variable or Constant	Type	Purpose
name	int	Persons name

Data Table for setAddress()

Variable or Constant	Type	Purpose
address	String	Persons address

Data Table for setPhoneNumber()

Variable or Constant	Type	Purpose
phoneNumber	String	Persons number

Data Table for setEmail()

Variable or Constant	Type	Purpose
email	String	Persons email

Data Table for setDate()

Variable or Constant	Type	Purpose
date	Date	A critical date

Data Table for toString()

Variable or Constant	Type	Purpose
person	String	The object as a string

Joseph Maples CS101
Design for an object-oriented people database

Data Table for Class Student

Variable or Constant	Type	Purpose
status	Date	Students status

Data Table for Student(name:String, address:String, phoneNumber:String, email:String, date:Date)

Variable or Constant	Type	Purpose
name	String	Persons name
address	String	Persons address
phoneNumber	String	Persons number
email	String	Persons email
birthDate	Date	A critical date
status	char	Students status

Data Table for setStatus()

Variable or Constant	Type	Purpose
status	char	Students status

Data Table for toString()

Variable or Constant	Type	Purpose
student	String	The object as a string
statusString	String	Status as a full word

Joseph Maples CS101
Design for an object-oriented people database

Data Table for UndergraduateStudent(name:String, address:String, phoneNumber:String, email:String, date:Date, status:char)

<u>Variable or Constant</u>	<u>Type</u>	<u>Purpose</u>
name	String	Persons name
address	String	Persons address
phoneNumber	String	Persons number
email	String	Persons email
birthDate	Date	A critical date
status	char	Students status

Data Table for toString()

<u>Variable or Constant</u>	<u>Type</u>	<u>Purpose</u>
student	String	The object as a string

Joseph Maples CS101
Design for an object-oriented people database

Data Table for Class GraduateStudent

Variable or Constant	Type	Purpose
assistantshipType	char	Students assistantship

Data Table for GraduateStudent(name:String, address:String, phoneNumber:String, email:String, date:Date, status:char. AssistantshipType:char)

Variable or Constant	Type	Purpose
name	String	Persons name
address	String	Persons address
phoneNumber	String	Persons number
email	String	Persons email
birthDate	Date	A critical date
status	char	Students status
assistantshipType	char	Students assistantship

Data Table for setAssistantshipType(assistantshipType:String)

Variable or Constant	Type	Purpose
assistantshipType	String	Students assistantship

Data Table for toString()

Variable or Constant	Type	Purpose
student	String	The object as a string
assistantshipString	String	Students assistantship as a full word

Joseph Maples CS101
Design for an object-oriented people database

Data Table for Class Employee

Variable or Constant	Type	Purpose
title	String	Job title
office	String	The office they work in
salary	double	Employee’s salary

Data Table for Employee(name:String, address:String, phoneNumber:String, email:String, date:Date, title:String, office:String, salary:double)

Variable or Constant	Type	Purpose
name	String	Persons name
address	String	Persons address
phoneNumber	String	Persons number
email	String	Persons email
hiringDate	Date	Date employee was hired
title	String	Job title
office	String	The office they work in
salary	double	Employee’s salary

Data Table for setTitle()

Variable or Constant	Type	Purpose
title	String	Job title

Data Table for setOffice()

Variable or Constant	Type	Purpose
office	String	The office they work in

Data Table for setSalary()

Variable or Constant	Type	Purpose
salary	double	Employee’s salary

Data Table for toString()

Variable or Constant	Type	Purpose
employee	String	The object as a string
money	NumberFormat	To format the salary

Joseph Maples CS101
Design for an object-oriented people database

Data Table for Class Faculty

Variable or Constant	Type	Purpose
officeHours	String	Hours in office

Data Table for Faculty(name:String, address:String, phoneNumber:String, email:String, date:Date, title:String, office:String, salary:double, officeHours:String)

Variable or Constant	Type	Purpose
name	String	Persons name
address	String	Persons address
phoneNumber	String	Persons number
email	String	Persons email
hiringDate	Date	Date employee was hired
title	String	Job title
office	String	The office they work in
salary	double	Employee’s salary
officeHours	String	Hours in office

Data Table for setOfficeHours()

Variable or Constant	Type	Purpose
officeHours	String	Hours in office

Data Table for toString()

Variable or Constant	Type	Purpose
faculty	String	The object as a string

Joseph Maples CS101
Design for an object-oriented people database

Data Table for Class Staff

Variable or Constant	Type	Purpose
supervisor	String	Employee’s supervisor

Data Table for Staff(name:String, address:String, phoneNumber:String, email:String, date:Date, title:String, office:String, salary:double, supervisor:String)

Variable or Constant	Type	Purpose
name	String	Persons name
address	String	Persons address
phoneNumber	String	Persons number
email	String	Persons email
hiringDate	Date	Date employee was hired
title	String	Job title
office	String	The office they work in
salary	double	Employee’s salary
supervisor	String	Employee’s supervisor

Data Table for setSupervisor()

Variable or Constant	Type	Purpose
supervisor	String	Employee’s supervisor

Data Table for toString()

Variable or Constant	Type	Purpose
staff	String	The object as a string

Joseph Maples CS101
Design for an object-oriented people database

Date Algorithms

```
toString()
    String date
    switch (month) :
      case 1:
        date equals "January "
        break
      case 2:
        date equals "February "
        break
      case 3:
        date equals "March "
        break
      case 4:
        date equals "April "
        break
      case 5:
        date equals "May "
        break
      case 6:
        date equals "June "
        break
      case 7:
        date equals "July "
        break
      case 8:
        date equals "August "
        break
      case 9:
        date equals "September "
        break
    case 10:
        date equals "October "
        break
    case 11:
        date equals "November "
        break
    case 12:
        date equals "December "
        break
    default:
        return "Invalid month!"
date += day + ", " + year
return date
```

Joseph Maples CS101
Design for an object-oriented people database

Driver Class Algorithms

```
main(String[] args) throws IOException
    File inFile equals new File(args[0])
    Scanner fileScan equals new Scanner(inFile)
    File outFile equals new File(args[1])
    PrintStream printer equals new PrintStream(outFile)
    Scanner getLines equals new Scanner(inFile)
    lines equals 0
    while (getLines.hasNextLine())
      lines+ 1
      getLines.nextLine()
    Person[] people equals Person[lines]
    index equals 0
    print to out file ("Project 6")
    print to out file ("Joseph Maples, Computing & Algorithms CS 101-02")
    print to out file ("The next group of outlines is an echo print of the input file\n")
    while (fileScan.hasNextLine())
      String line equals fileScan.nextLine()
      print to out file (line)
      String[] data equals line.split("#")
      switch (data[0].charAt(0)) :
        case 'u':
          people[index] equals new UndergraduateStudent(data[1], data[2], data[3], data[4], new Date(data[5]), data[6].charAt(0))
          index+ 1
          break
        case 'g':
          people[index] equals new GraduateStudent(data[1], data[2], new Date(data[3]), data[4], data[5], data[6].charAt(0), data[7].charAt(0))
          index+ 1
          break
        case 'f':
          people[index] equals new Faculty(data[1], data[2], data[3], data[4], data[5], Double.parseDouble(data[6]), new Date(data[7]), data[8], dat
                index+ 1
                break
            case 's':
                people[index] equals new Staff(data[1], data[2], data[3], data[4], data[5], data[6], Double.parseDouble(data[7]), new Date(data[8]), data[
                index+ 1
                break
        print to out file ("")
        print to out file ("Entire database, sorted by name")
        sortByName(people)
        for index equals 0 loop till index is less than people.length by index+ 1 each step
            print to out file (people[index].toString())
        print to out file ("Entire staff, sorted by date")
        sortByDate(people)
        for index equals 0 loop till index is less than people.length by index+ 1 each step
            if (people[index] instanceof Staff)
                print to out file (people[index].toString())
        print to out file ("Every Employee, sorted by salary")
        sortBySalary(people)
        for index equals 0 loop till index is less than people.length by index+ 1 each step
            if (people[index] instanceof Employee)
                print to out file (people[index].toString())
        print to out file ("Every Graduate student, sorted by address")
        sortByAddress(people)
        for index equals 0 loop till index is less than people.length by index+ 1 each step
            if (people[index] instanceof GraduateStudent)
                print to out file (people[index].toString())

sortByName(Person[] array)
        size equals array.length
        for index equals 0 loop till index is less than size - 1 by index+ 1 each step
            minIndex equals index
            for index2 equals index + 1 loop till index2 is less than size by index2+ 1 each step
                if (array[index2].getName().compareTo(array[minIndex].getName()) is less than 0)
                    minIndex equals index2
            Person temp equals array[minIndex]
            array[minIndex] equals array[index]
            array[index] equals temp

sortByDate(Person[] array)
        size equals array.length
        for index equals 0 loop till index is less than size - 1 by index+ 1 each step
            minIndex equals index
            for index2 equals index + 1 loop till index2 is less than size by index2+ 1 each step
                if (array[index2].getDate().compareTo(array[minIndex].getDate()) is less than 0)
                    minIndex equals index2
            Person temp equals array[minIndex]
            array[minIndex] equals array[index]
            array[index] equals temp

sortBySalary(Person[] array)
        [] employees equals employeeIndicies(array)
        for index equals 0 loop till index is less than employees.length - 1 by index+ 1 each step
            minIndex equals employees[index]
            for index2 equals index + 1 loop till index2 is less than employees.length by index2+ 1 each step
                if (((Employee) array[employees[index2]]).getSalary() is less than ((Employee) array[minIndex]).getSalary())
                    minIndex equals employees[index2]
            Person temp equals array[minIndex]
            array[minIndex] equals array[employees[index]]
            array[employees[index]] equals temp

sortByAddress(Person[] array)
        size equals array.length
        for index equals 0 loop till index is less than size - 1 by index+ 1 each step
            minIndex equals index
            for index2 equals index + 1 loop till index2 is less than size by index2+ 1 each step
                if (array[index2].getAddress().compareTo(array[minIndex].getAddress()) is less than 0)
                    minIndex equals index2
            Person temp equals array[minIndex]
            array[minIndex] equals array[index]
            array[index] equals temp
```

```
employeeIndicies(Person[] array)
    String indexString equals ""
    for index equals 0 loop till index is less than array.length by index+ 1 each step
        if (array[index] instanceof Employee)
            indexString += index + ","
    String[] indicies equals indexString.split(",")
    [] employees equals [0..indicies.length-1]
    for element equals 0 loop till element is less than indicies.length by element+ 1 each step
        employees[element] equals Integer.parseInt(indicies[element])
    return employees
```

:a[9])

9])

Joseph Maples CS101
Design for an object-oriented people database

Driver Class Algorithms

```
toString()
    String person
    person equals "\tname: " + name + "\n"
    person += "\taddress: " + address + "\n"
    person += "\tphone number: " + phoneNumber + "\n"
    person += "\temail: " + email + "\n"
    return person
```

Joseph Maples CS101
Design for an object-oriented people database

Student Class Algorithms

```
toString()
    String student
    String statusString
    switch (status) :
      case 'f':
        statusString equals "freshmen"
        break
      case 's':
        statusString equals "sophomore"
        break
      case 'j':
        statusString equals "junior"
        break
      case 'r':
        statusString equals "senior"
        break
      case 'm':
        statusString equals "masters"
        break
      case 'd':
        statusString equals "doctorate"
        break
      default:
        statusString equals "Invalid status!"
    student equals super.toString()
    student += "\tbirth date: " + date.toString() + "\n"
    student += "\tstatus: " + statusString + "\n"
    return student
```

Joseph Maples CS101
Design for an object-oriented people database

UndergraduateStudent Class Algorithms

```
toString()
    String undergrad equals "Undergraduate Student\n"
    undergrad += super.toString()
    return undergrad
```

Joseph Maples CS101
Design for an object-oriented people database

GraduateStudent Class Algorithms

```
toString()
    String assistantshipString
    switch (assistantshipType) :
      case 't':
        assistantshipString equals "teaching"
        break
      case 'r':
        assistantshipString equals "research"
        break
      default:
        assistantshipString equals "Invalid status!"
    String graduate equals "Graduate student\n"
    graduate += super.toString()
    graduate += "\tassistantship type: " + assistantshipString + "\n"
    return graduate
```

Joseph Maples CS101
Design for an object-oriented people database

Employee Class Algorithms

```
toString()
    NumberFormat money equals NumberFormat.getCurrencyInstance()
    String employee
    employee equals super.toString()
    employee += "\ttitle: " + title + "\n"
    employee += "\toffice: " + office + "\n"
    employee += "\tsalary: " + money.format(salary) + "\n"
    employee += "\thiring date: " + date.toString() + "\n"
    return employee
```

Joseph Maples CS101
Design for an object-oriented people database

Staff Class Algorithms

```
toString()
    String staff equals "Staff\n"
    staff += super.toString()
    staff += "\tsupervisor: " + supervisor + "\n"
    return staff
```

Joseph Maples CS101
Design for an object-oriented people database

Faculty Class Algorithms

```
toString()
    String faculty equals "Faculty\n"
    faculty += super.toString()
    faculty += "\toffice hours: " + officeHours + "\n"
    return faculty
```