James Henson

HW1

3.1

ArrayList<Integer> list = new ArrayList<>();

for (int i=0; i < 5; i++){

list.add(i);}

for (int i=0; i < 5; i++){

list.add(i);}

for (int i : list){

System.out.print(i);}

3.2 After the loop completes list contains = {1, 3, 6, 10, 15, 19, 22, 24, 25}

3.3

int count = 0;

for (int i : list){

if(i < 0) count++;}

System.out.println("Count Negative: " +count);

3.4

Public int arrayListSum(ArrayList<Integer> pList){

int sum = 0;

for (int i = 0; i < pList.size(); i++){

sum += pList.get(i);}

return sum;

3.5

Public int arrayListCreate(int pLen, int pInitValue){

ArrayList<Integer> list = new ArrayList<>(pLen);

for (int i = 0; i < pLen; i++){

list.set(i, pInitValue);

} return list;

}

3.6

public ArrayList<String> insertName(ArrayList<String> pList, String pName){

int pos = 0;

for (int i = 0; i < pList.size(); i++){

if (pList.get(i).charAt(0) > pName.charAt(0)){ //find position

pos = i;

break;

}

}

for (int i = pos +1; i < pList.size()+1; i++){ //increase list size for added name

pList.set(i, pList.get(i-1));}

pList.set(pos, pName);

return pList;

}

3.7

public ArrayList<Integer> arrayListRemove(ArrayList<Integer> pList, int pValue){

for (int i = 0; i < pList.size(); i++) {

if (pList.get(i) == pValue){

pList.remove(i);}

}

4.1 java.io.FileNotFoundException is thrown.

4.2 IOException error is thrown again by the system.

5.1 Throwing an exception in a method sends the exception up the proverbial chain until the Java Virtual Machine gets the exception and terminates the program and prints a stack trace. Catching an exception can be seen when using a try/catch block and allows the code author to include code to execute if a specific exception occurs.

5.2 A checked exception is an exception where the Java compiler will check to see if the code author has included some form of exception handling for the specific exception. One instance if a file not found exception. In this case (checked exception), the compiler will throw an exception or the author can write an exception handler:

try{

//some code that throws file not found exception

} catch (FileNotFoundException e) { e.printStackTrace();}

5.3 Unchecked errors occur when a bug is present in the code. This causes the JVM to terminate the program and the bug should be investigated. A common unchecked error is attempting to divide by zero.

5.4 Checked Exceptions

5.5 IndexOutOfBoundsException is a runtime exception that will be thrown.

5.6 Yes, the exception object is always the same as the type declared in the catch clause. When a checked exception occurs, JVM looks for an exception handler with the exact name.

5.7 Finally clause is used to terminate anything necessary as well as code that needs to be executed. In the case of this assignment (4.3), a finally block was used to close the BufferedReader and BufferedWriter streams.

5.8 nextInt(): InputMismatchException and next(): NoSuchElementException are both checked exceptions that are thrown.

6.1 Instance method applies to an object (i.e. instance of the class) that operate on instance and class variables. Meanwhile, a class method operates of class variables and are used to create instances of class.

6.2 Compiler inserts default constructor.

6.3 (a) An instance method must instantiated before acting upon an object. With a static method, no instance of the class is needed to run the method. (b) Methods from an instance method must first be instantiated.

6.4 NullPointerException causes program to fail. The instance variable s is instantiated and has a value of null. When called in s.length(), error is thrown.

6.5

public class C {

public static final int A = 100;

public static final int B = 200;

private int mX;

private int mY;

public C() {

mX= -1;

}

public C(int pX) {

setX(pX);

}

public int getX() {

return mX;

}

public int getY() {

return mY;

}

public void setX(int pX) {

mX = pX;

}

public void setY(int pY) {

mY = pY;

}

}

6.6

public static void main(String[] args){

C cObj1 = new C();

}

6.7

C cObj2 = new C(10);

6.8

a) mX is private so it will not complie – cannot be reached by outside class

b) mY is private so it will not compile – cannot be reached by outside class

c) It is public and complies to a3 = 100

d) It is public and a4 = 200

e) Will not compile – method is undefined

f) Compiles as a5 = -1

g) Legal, sets value of mX to 20

h) Legal, sets value of mX to -1

i) Does not compile – cannot access class method by class name

j) Does not compile – class method cannot be accessed in that format

k) Legal, sets a7 = 0

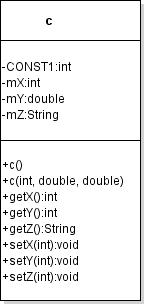
l) Legal, sets mY – 20

m) Illegal, getY is class method, not static

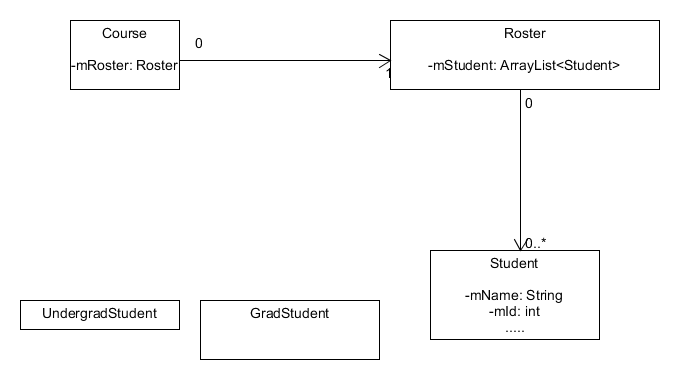
n) Illegal, setY is class method, not static

6.9 f() is legal and sets mX = 0 and mY = 3. g() is not legal, cannot use mY in static method.

7.1



7.2



a)