CSE115 / CSE503
Introduction to Computer Science I

Dr. Carl Alphonce
343 Davis Hall
alphonce@buffalo.edu

Office hours:
Thursday 12:00 PM – 2:00 PM
Friday 8:30 AM – 10:30 AM
OR request appointment via e-mail
Turn off and put away electronics:

- cell phones
- pagers
- laptops
- tablets
- etc.
Today

Collections
Control structures

Coming up

More collections/control structures
Circling back to inheritance and constructor chaining
FINAL EXAM SCHEDULE

Arranged by the university
Check HUB for date/time/room

UNCLAIMED EXAMS

Pick up at end of class today
COLLECTIONS & CONTROL STRUCTURES
channel 1
I picked up my exam on Friday

A. Yes
B. No
Question 4
Define a class LipoProteinPanel so that a LipoProteinPanel object stores the results of a cholesterol test: total cholesterol, LDL cholesterol and HDL cholesterol. Each of these individual results is given by an int value. Don't forget to define a constructor that lets these three values be provided as arguments, as in:

```java
LipoProteinPanel p = new LipoProteinPanel(190, 90, 74);
```

The order of arguments must be total cholesterol, LDL cholesterol and HDL cholesterol. Finally, define a no-argument method named 'normal' that returns true if total cholesterol is less than 200 and LDL cholesterol is less than 130 and HDL cholesterol is greater than or equal to 40. The method must return false otherwise.
Define a class \texttt{LipoProteinPanel} so that a \texttt{LipoProteinPanel} object stores the results of a cholesterol test: total cholesterol, LDL cholesterol and HDL cholesterol. Each of these individual results is given by an \texttt{int} value. Don't forget to define a constructor that lets these three values be provided as arguments, as in:

\begin{verbatim}
LipoProteinPanel p = new LipoProteinPanel(190, 90, 74);
\end{verbatim}

The order of arguments must be total cholesterol, LDL cholesterol and HDL cholesterol. Finally, define a no-argument method named 'normal' that returns true if total cholesterol is less than 200 and LDL cholesterol is less than 130 and HDL cholesterol is greater than or equal to 40. The method must return false otherwise.
public class LipoProteinPanel {

    private int _total;
    private int _ldl;
    private int _hdl;

    public LipoProteinPanel(int tot, int ldl, int hdl) {
        _total = tot;
        _ldl = ldl;
        _hdl = hdl;
    }

    public boolean normal() {
        return _total < 200 && _ldl < 130 && _hdl >= 40;
    }

}
Define a class `LipoProteinPanel` so that a `LipoProteinPanel` object stores the results of a cholesterol test: total cholesterol, LDL cholesterol and HDL cholesterol. Each of these individual results is given by an `int` value. Don't forget to define a constructor that lets these three values be provided as arguments, as in:

```java
LipoProteinPanel p = new LipoProteinPanel(190, 90, 74);
```

The order of arguments must be total cholesterol, LDL cholesterol and HDL cholesterol. Finally, define a no-argument method named 'normal' that returns true if total cholesterol is less than 200 and LDL cholesterol is less than 130 and HDL cholesterol is greater than or equal to 40. The method must return false otherwise.
public class LipoProteinPanel {

    private int _total;
    private int _ldl;
    private int _hdl;

    public LipoProteinPanel(int tot, int ldl, int hdl) {
        _total = tot;
        _ldl = ldl;
        _hdl = hdl;
    }

    public boolean normal() {
        return _total < 200 && _ldl < 130 && _hdl >= 40;
    }

}
Define a class LipoProteinPanel so that a LipoProteinPanel object stores the results of a cholesterol test: total cholesterol, LDL cholesterol and HDL cholesterol. Each of these individual results is given by an int value. Don't forget to define a constructor that lets these three values be provided as arguments, as in:

```
LipoProteinPanel p = new LipoProteinPanel(190, 90, 74);
```

The order of arguments must be total cholesterol, LDL cholesterol and HDL cholesterol. Finally, define a no-argument method named 'normal' that returns true if total cholesterol is less than 200 and LDL cholesterol is less than 130 and HDL cholesterol is greater than or equal to 40. The method must return false otherwise.
public class LipoProteinPanel {

    private ??? _total;
    private ??? _ldl;
    private ??? _hdl;

    public LipoProteinPanel(int tot, int ldl, int hdl) {
        _total = tot;
        _ldl = ldl;
        _hdl = hdl;
    }

    public boolean normal() {
        return _total < 200 && _ldl < 130 && _hdl >= 40;
    }
}
Define a class LipoProteinPanel so that a LipoProteinPanel object stores the results of a cholesterol test: total cholesterol, LDL cholesterol and HDL cholesterol. Each of these individual results is given by an int value. Don't forget to define a constructor that lets these three values be provided as arguments, as in:

```
LipoProteinPanel p = new LipoProteinPanel(190, 90, 74);
```

The order of arguments must be total cholesterol, LDL cholesterol and HDL cholesterol. Finally, define a no-argument method named 'normal' that returns true if total cholesterol is less than 200 and LDL cholesterol is less than 130 and HDL cholesterol is greater than or equal to 40. The method must return false otherwise.
```java
public class LipoProteinPanel {

    private int _total;
    private int _ldl;
    private int _hdl;

    public LipoProteinPanel(int tot, int ldl, int hdl) {
        _total = tot;
        _ldl = ldl;
        _hdl = hdl;
    }

    public boolean normal() {
        return _total < 200 && _ldl < 130 && _hdl >= 40;
    }
}
```
Define a class LipoProteinPanel so that a LipoProteinPanel object stores the results of a cholesterol test: total cholesterol, LDL cholesterol and HDL cholesterol. Each of these individual results is given by an int value. Don't forget to define a constructor that lets these three values be provided as arguments, as in:

```java
LipoProteinPanel p = new LipoProteinPanel(190, 90, 74);
```

The order of arguments must be total cholesterol, LDL cholesterol and HDL cholesterol. Finally, define a no-argument method named 'normal' that returns true if total cholesterol is less than 200 and LDL cholesterol is less than 130 and HDL cholesterol is greater than or equal to 40. The method must return false otherwise.
public class LipoProteinPanel {

    private int _total;
    private int _ldl;
    private int _hdl;

    public LipoProteinPanel(int tot, int ldl, int hdl) {
        _total = tot;
        _ldl = ldl;
        _hdl = hdl;
    }

    public boolean normal() {
        return _total < 200 && _ldl < 130 && _hdl >= 40;
    }
}

© Dr. Carl Alphonce
Define a class LipoProteinPanel so that a LipoProteinPanel object stores the results of a cholesterol test: total cholesterol, LDL cholesterol and HDL cholesterol. Each of these individual results is given by an int value. Don't forget to define a constructor that lets these three values be provided as arguments, as in:

```java
LipoProteinPanel p = new LipoProteinPanel(190, 90, 74);
```

The order of arguments must be total cholesterol, LDL cholesterol and HDL cholesterol. Finally, define a no-argument method named 'normal' that returns true if total cholesterol is less than 200 and LDL cholesterol is less than 130 and HDL cholesterol is greater than or equal to 40. The method must return false otherwise.
public class LipoProteinPanel {

    private int _total;
    private int _ldl;
    private int _hdlg;

    public LipoProteinPanel(int tot, int ldl, int hdl) {
        _total = tot;
        _ldl = ldl;
        _hdlg = hdl;
    }

    public boolean normal() {
        return _total < 200 && _ldl < 130 && _hdlg >= 40;
    }
}
public class LipoProteinPanel {

    private int _total;
    private int _ldl;
    private int _hdl;

    public LipoProteinPanel(int tot, int ldl, int hdl) {
        _total = tot;
        _ldl = ldl;
        _hdl = hdl;
    }

    public boolean normal() {
        return _total < 200 && _ldl < 130 && _hdl >= 40;
    }

}
Question 5
Assume that the LipoProteinPanel class discussed in question 4 has been correctly defined. Define a method named flag that takes an ArrayList\(^{<}\)LipoProteinPanel\(^{>}\) and returns a new ArrayList\(^{<}\)LipoProteinPanel\(^{>}\). The returned ArrayList must contain all the LipoProteinPanel objects in the argument list for which the normal method returns false. For example:

```java
ArrayList\(<\)LipoProteinPanel\(>\) input;
input = new ArrayList\(<\)LipoProteinPanel\(>\)();

LipoProteinPanel a = new LipoProteinPanel(190,90,74);
LipoProteinPanel b = new LipoProteinPanel(205,90,74));
LipoProteinPanel c = new LipoProteinPanel(190,141,74));

input.add(a);
input.add(b);
input.add(c);

ArrayList\(<\)LipoProteinPanel\(>\) actual = flag(input);
```

Actual must contain exactly b and c. In particular, actual must not contain a.
Assume that the LipoProteinPanel class discussed in question 4 has been correctly defined. Define a method named flag that takes an ArrayList<LipoProteinPanel> and returns a new ArrayList<LipoProteinPanel>. The returned ArrayList must contain all the LipoProteinPanel objects in the argument list for which the normal method returns false. For example:

```java
ArrayList<LipoProteinPanel> input;
input = new ArrayList<LipoProteinPanel>();

LipoProteinPanel a = new LipoProteinPanel(190,90,74);
LipoProteinPanel b = new LipoProteinPanel(205,90,74);
LipoProteinPanel c = new LipoProteinPanel(190,141,74);

input.add(a);
input.add(b);
input.add(c);

ArrayList<LipoProteinPanel> actual = flag(input);
```

Actual must contain exactly b and c. In particular, actual must not contain a.
Assume that the LipoProteinPanel class discussed in question 4 has been correctly defined. Define a method named flag that takes an ArrayList<LipoProteinPanel> and returns a new ArrayList<LipoProteinPanel>. The returned ArrayList must contain all the LipoProteinPanel objects in the argument list for which the normal method returns false. For example:

```java
ArrayList<LipoProteinPanel> input;
input = new ArrayList<LipoProteinPanel>();

LipoProteinPanel a = new LipoProteinPanel(190,90,74);
LipoProteinPanel b = new LipoProteinPanel(205,90,74);
LipoProteinPanel c = new LipoProteinPanel(190,141,74);

input.add(a);
input.add(b);
input.add(c);

ArrayList<LipoProteinPanel> actual = flag(input);
```

Actual must contain exactly b and c. In particular, actual must not contain a.
public ArrayList<LipoProteinPanel> flag(ArrayList<LipoProteinPanel> results) {

    ArrayList<LipoProteinPanel> flaggedResults = new ArrayList<LipoProteinPanel>();

    for (LipoProteinPanel p : results) {
        if (!p.normal()) {
            flaggedResults.add(p);
        }
    }

    return flaggedResults;
}
Assume that the LipoProteinPanel class discussed in question 4 has been correctly defined. Define a method named flag that takes an ArrayList<LipoProteinPanel> and returns a new ArrayList<LipoProteinPanel>. The returned ArrayList must contain all the LipoProteinPanel objects in the argument list for which the normal method returns false. For example:

```java
ArrayList<LipoProteinPanel> input;
input = new ArrayList<LipoProteinPanel>();

LipoProteinPanel a = new LipoProteinPanel(190, 90, 74);
LipoProteinPanel b = new LipoProteinPanel(205, 90, 74);
LipoProteinPanel c = new LipoProteinPanel(190, 141, 74);

input.add(a);
input.add(b);
input.add(c);

ArrayList<LipoProteinPanel> actual = flag(input);
```

Actual must contain exactly b and c. In particular, actual must not contain a.
public ArrayList<LipoProteinPanel> flag(ArrayList<LipoProteinPanel> results) {
    ArrayList<LipoProteinPanel> flaggedResults;
    for (LipoProteinPanel p : results) {
        if (!p.normal()) {
            flaggedResults.add(p);
        }
    }
    return flaggedResults;
}
public ArrayList<LipoProteinPanel> flag(ArrayList<LipoProteinPanel> results) {

    ArrayList<LipoProteinPanel> flaggedResults;

    flaggedResults = new ArrayList<LipoProteinPanel>();

    for (LipoProteinPanel p : results) {
        if (!p.normal()) {
            flaggedResults.add(p);
        }
    }

    return flaggedResults;
}
public ArrayList<LipoProteinPanel> flag(ArrayList<LipoProteinPanel> results) {

    ArrayList<LipoProteinPanel> flaggedResults;

    flaggedResults = new ArrayList<LipoProteinPanel>();

    for (LipoProteinPanel p : results) {

    }

    return flaggedResults;

}
public ArrayList<LipoProteinPanel> flag(ArrayList<LipoProteinPanel> results) {

    ArrayList<LipoProteinPanel> flaggedResults;
    flaggedResults = new ArrayList<LipoProteinPanel>();

    for (LipoProteinPanel p : results) {
        if (!p.normal()) {
            flaggedResults.add(p);
        }
    }

    return flaggedResults;
}
public ArrayList<LipoProteinPanel> flag(ArrayList<LipoProteinPanel> results) {

    ArrayList<LipoProteinPanel> flaggedResults;

    flaggedResults = new ArrayList<LipoProteinPanel>();

    for (LipoProteinPanel p : results) {

        if (!p.normal()) {

            flaggedResults.add(p);
        }
    }

    return flaggedResults;
}
public ArrayList<LipoProteinPanel> flag(ArrayList<LipoProteinPanel> results) {

    ArrayList<LipoProteinPanel> flaggedResults = new ArrayList<>(null);

    for (LipoProteinPanel p : results) {
        if (!p.normal()) {
            flaggedResults.add(p);
        }
    }

    return flaggedResults;
}
channel 1
How is this written as a regular for loop?

```java
for (LipoProteinPanel p : results) {
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}
```

A. for (int i=0; i<results.size()-1; i=i+1) {
    LipoProteinPanel p = results.get(i);
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}

B. for (int i=1; i<results.size(); i=i+1) {
    LipoProteinPanel p = results.next();
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}

C. for (int i=0; i<results.size(); i=i+1) {
    LipoProteinPanel p = results.get(i);
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}
How is this written as a regular for loop?

```java
for (LipoProteinPanel p : results) {
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}
```

A. for (int i=0; i<results.size()-1; i=i+1) {
    LipoProteinPanel p = results.get(i);
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}

B. for (int i=1; i<results.size(); i=i+1) {
    LipoProteinPanel p = results.next();
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}

C. for (int i=0; i<results.size(); i=i+1) {
    LipoProteinPanel p = results.get(i);
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}
```

Convince your neighbor your answer is correct!
How is this written as a regular for loop?

```java
for (LipoProteinPanel p : results) {
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}
```

A. for (int i=0; i<results.size()-1; i=i+1) {
    LipoProteinPanel p = results.get(i);
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}

B. for (int i=1; i<results.size(); i=i+1) {
    LipoProteinPanel p = results.next();
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}

C. for (int i=0; i<results.size(); i=i+1) {
    LipoProteinPanel p = results.get(i);
    if (!p.normal()) {
        flaggedResults.add(p);
    }
}

```