Announcement

- The deadline for Lab 5 is extended to Sunday, March 24\textsuperscript{th}
- The final project – Program 4 will be online soon, hopefully in this week
Midterm Exam

- Total points on the last page
- Points for each part at the beginning of each part
- Points for each question near the question
Overall Grade

• Average: 71

• Distribution:
  – Higher than 90: 7
  – Between 80 and 90: 9
  – Between 70 and 80: 6
  – Between 60 and 70: 5
  – Between 50 and 60: 7
  – Below 50: 8
The Objective of the Exam

• You have to summarize: what part to improve?
• The final exam will be in the same form
  – Will be a little easier, but more questions are expected
    • It will take 3 hours!
  – Prepare yourself for the final exam
    • The midterm is only 10% of the whole course
    • The final is 25% of the whole course
• Please talk to me if you feel that the grade is not reflecting your effort
  – There may be something wrong in your learning style
Key Skills

• How to solve a problem in general
  – Write pseudocode for your algorithm!
    • You get points for pseudocode

• How to express the solutions in Java
  – Especially, without the help of textbook, lecture notes, past assignments, and Eclipse
Question 1-4

- Easy questions (shouldn’t lose points)
  - Nothing special
  - Just read the solutions
Question 5

• Easy question
  – `bestFriend` and `FIFTYSEVEN57` are fine
  – `7daysAWeek` starts with a number
  – `hello!` and `TOTAL&COST` include special characters
    • Remember that `!` and `&&` have meanings
  – `private`, `do` and `new` are keywords in Java
Question 6

• Mid-level question

• `double var1 = 10 / 3;`
  – The answer is 3
  – The right side is an integer, because 10 and 3 are both int

• `int var2 = (int) (2.5 * 2.6);`
  – The answer is 6
  – 2.5*2.6 = 6.5. Then 6.5 is converted to integer
Question 6

• \texttt{boolean var3 = !(3 > 3);}
  
  — True
  
  — 3>3 is false. The negative will be true

• \texttt{boolean var4 = (121 \% 11 == 0) || (5 == 5);}
  
  — True
  
  — 121 \% 11 is 0. You can decide it’s true because 5 == 5.

• \texttt{int var5 = 11 \% 3;}
  
  — 11 = 3*3 +2. The answer is 2.
Question 7

• Easy question

• *double accountBalance = 245.25;*
  
  – Can’t use int!
Question 8

• Supposed to be easy, but turned out to be mid-level

• $a = b$
  – It is an assignment statement
  – Let a have b’s value

• $a == b$
  – It is a boolean expression, the value depends on a and b

• $a += b$
  – $a = a + b$, add the value of b to a
  – Not “add a to b”!!!
Question 9

• Easy question

How are you?

0 1 2 3 4 5 6 7 8 9 10 11
Question 10

- Easy question
- `str.length()`
  - int type, the value is 12, not 11
- `str.equalsIgnoreCase("HOW ARE YOU")`
  - boolean type. The value can only be true or false
  - Think about `str.equals(anotherString)`
  - The answer is false, because the last ‘?’ is missing.
  - `str.equalsIgnoreCase("HOW ARE YOU?")` will be true
Question 10

- `str.indexOf("ou")`
  - int type, the value is 9
  - The value is not 9 and 10
  - An integer can not have two values
  - `indexOf()` can search for a single character, or a string
  - The first position where “ou” appears is 9

```
 How are you?
```

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>y</th>
<th>o</th>
<th>u</th>
<th>?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
</tbody>
</table>
Question 10

• `str.lastIndexOf(" ")`
  – int type, the value is 7
• `str.charAt(6)`
  – char type, the value is ‘e’
• `str.substring(1,6)`
  – String type, the value is “ow ar”
Question 11

• Supposed to be easy, turn out to be mid-level

```java
int count = 0;
while (count < 100)
    if (count % 5 == 0)
        System.out.println(count);
    count++;
```

• count++ isn’t in the loop. The loop will never end
Question 11

• Correct code

```java
int count = 0;
while (count < 100) {
    if (count % 5 == 0) {
        System.out.println(count);
    }
    count++;
}
```

– This piece of code will print all numbers that are smaller than 100 and can be divided by 5

• 0, 5, 10, 15, ...., 95
Question 11

- Someone thought the code was

```java
int count = 0;
while (count < 100) {
    if (count % 5 == 0) {
        System.out.println(count);
        count++;
    }
}
```

- Then it will also be an infinite loop because if count = 1, it will not be increased

- This answer was counted as correct
• Supposed to be easy, turned out to be hard!

do {
    System.out.print("The programming language, "Java", ");
} while (false);
System.out.println("is named after the Java coffee");

– The do-while loop is legal!
– It will start from the body, print the first string
– Then because the condition expression is false, it will not repeat the loop, and continue to print the second string
– It will be an infinite loop if changed to be while(true)!
Question 12

• The problem is about the quotation marks

```java
do {
    System.out.print("The programming language, "Java", ");
} while (false);
System.out.println("is named after the Java coffee");
```

– “Java” is outside of the paired quotation marks
– This is a syntax error
Question 12

• The correct code

```java
do {
    System.out.print("The programming language, \"Java\", ");
} while (false);
System.out.println("is named after the Java coffee");
```

– We use backslash symbol to include quotation marks in a string

– The correct code will print:
  • The programming language, "Java", is named after the Java coffee

– And it is true
Question 13

• Easy question

```java
public int absoluteValue(int num) {
    if (num < 0);
        return -num;
    else if (num > 0)
        return num;
}
```

– The first semicolon is wrong. It will end the whole if statement
– The case that num is 0 is not covered
Question 13

• Correct code

```java
public int absoluteValue(int num) {
    if (num < 0) {
        return -num;
    } else {
        return num;
    }
}
```

– This piece of code appeared on Lecture 13, the review session!

– Someone thought -num is wrong
  • It is correct. It represents the negative of a variable
Question 14

- Mid-level question

```java
public void swap(int a, int b) {
    int temp = a;
    a = b;
    b = temp;
}
```

- The swap() method won't change anything because all variables are local

```java
public void doSomething() {
    int a = 2, b = 3;
    a = b;
    b = a;
    System.out.println(a + "\," + b);
    int c = 2, d = 3;
    int temp = c;
    c = d;
    d = temp;
    System.out.println(c + "\," + d);
    int e = 2, f = 3;
    swap(f, e);
    System.out.println(e + "\," + f);
}
```
public void doSomething() {
    int a = 2, b = 3;
    a = b; // a is 3 now
    b = a; // b is assigned to be a, which is 3!
    System.out.println(a + "," + b); // print 3,3
    int c = 2, d = 3;
    int temp = c; // temp is 2 now
    c = d; // c is 3 now
    d = temp; // d is 2. This is how we swap variables
    // Think about if you swap liquid in two cups –
    // you need another cup to do that!
    System.out.println(c + "," + d); // print 3,2
    int e = 2, f = 3;
    swap(f, e); // All changes in swap() are local
    System.out.println(e + "," + f); // 2,3 – nothing changed
}
Question 15

• Supposed to be easy, turned out to be hard!

```java
int a = 2, b = 2, c = 2;
for (int i = 0; i < 3; i++) {
    a += a;
    b *= b;
    c /= c;
}
System.out.println(a + " , " + b + " , " + c + ".");
```

• \( a = a+a, b = b*b, c=c/c \)

• **Repeat 3 times!!!**
Question 15

- **The first time**
  - $a = a + a = 2 + 2 = 4$; $b = b \times b = 2 \times 2 = 4$; $c = c \div c = 2 \div 2 = 1$;

- **The second time**
  - $a = a + a = 4 + 4 = 8$; $b = b \times b = 4 \times 4 = 16$; $c = c \div c = 1 \div 1 = 1$;
  - Pay attention that the value of $a$, $b$, $c$ are **changed** after the first loop!

- **The third time**
  - $a = a + a = 8 + 8 = 16$; $b = b \times b = 16 \times 16 = 256$; $c = c \div c = 1 \div 1 = 1$;

- The result: 16, 256, 1
Question 16

• Mid-level question

```java
public void test(int k) {
    String t = "The quick brown fox jumps over the lazy dog";
    for (int i = 0; i < k; i++) {
        t = t.substring(t.indexOf(" ") + 1);
    }
    System.out.println(t.substring(0, t.indexOf(" ") ));
}
```

– Key point: what does this for loop do?
– **It deletes the first k words!**
Extract Words (From Lecture 10)

```java
String t = "2.5 + 3 + 5 + 12 + 16";
while (t.indexOf(" ") != -1) {
    String temp = t.substring(0, t.indexOf(" "));
    t = t.substring(t.indexOf(" ") + 1);
    System.out.print("**" + temp + "**");
}
System.out.println("**" + t + "**");
```

• This piece of code will extract each single word in the string
  – While there is at least one space, we print the first word

• The output will be
  **2.5**+**3**+**5**+**12**+**16**
Question 16

• Mid-level question

```java
public void test(int k) {
    String t = "The quick brown fox jumps over the lazy dog";
    for (int i = 0; i < k; i++) {
        t = t.substring(t.indexOf(" ") + 1);
    }
    System.out.println(t.substring(0, t.indexOf(" ")));    
}
```

– Reading the loop

• In each loop body, t is updated to be the string after its first " "
• Therefore, the first word is deleted
• After k times, the first k words are deleted
Question 16

• Mid-level question

```java
public void test(int k) {
    String t = "The quick brown fox jumps over the lazy dog";
    for (int i = 0; i < k; i++) {
        t = t.substring(t.indexOf(" ") + 1);
    }
    System.out.println(t.substring(0, t.indexOf(" "))); }
```

– After the loop, we print the substring from the beginning to the first “ ”
• We print the first word
Question 16

• Mid-level question

```java
public void test(int k) {
    String t = "The quick brown fox jumps over the lazy dog";
    for (int i = 0; i < k; i++) {
        t = t.substring(t.indexOf(" ") + 1);
    }
    System.out.println(t.substring(0, t.indexOf(" ")));}
```

– Therefore, for test(1), it will remove 1 word then print the first word in the remaining string, which is “quick”
– test(3) will print the 4th word, which is “fox”
– test(5) will print the 6th word, which is “over”
Question 17-19

• Supposed to be easy to mid-level, turned out to be mid-level (not so bad compared with 11-16)

• First problem: no pseudocode!
  – If you don’t write down your idea, I have to guess from your sketch code – it is really hard!
  – Also, it is hard for yourself to follow the whole logic flow

• Second problem: not familiar with methods
  – I used methods so that you don’t have to write complicated user interactions
    • You didn’t lose points for not using methods correctly
Question 17

• Supposed to be easy, turned out to be mid-level
• The requirement: count all divisors
• How do you count things?
  – You try all cases
  – For each case that fulfills the requirement, you add the total number by 1
Question 17

• Supposed to be easy, turned out to be mid-level
• The requirement: count all divisors
• How do you count things?
  – You try all cases
    • Try all cases: test every positive integer no greater than N
  – For each case that fulfills the requirement, you add the total number by 1
    • Use a variable to count the value. If the integer is a divisor, add the counting variable by 1
Question 17

• You try all cases
  – Try all cases: test every positive integer no greater than N
  – for (int i=1; i<=N; i++)

• For each case that fulfills the requirement, you add the total number by 1
  – Use a variable to count the value. If the integer is a divisor, add the counting variable by 1
  – if (N%i == 0) count++;
  – Remember to initialize count as 0
• Now it is a complete program

```java
public int divisors(int N) {
    int count = 0;
    for (int i = 1; i <= N; i++) {
        if (N % i == 0) {
            count++;
        }
    }
    return count;
}
```
Question 17

• Advanced version
  – The hint is quite straightforward: if \( N\%i == 0 \), then of course \( N\%(N/i) \) is also 0
  – Therefore, every time we find a divisor, there is another paired divisor. We can increase counter by 2
  – How to avoid over-counting? We only count the small value in the pair. Therefore, the loop stops at \( \text{Math.sqrt}(N) \)
  – The only problem: for \( \text{Math.sqrt}(N) \) itself, if it is an integer, there is no paired integer (think about \( 100\%10 == 0 \)). Therefore, we have to deal with this special case
Question 17

• Advanced version

```java
public int factors(int N) {
    int count = 0;
    for (int i = 1; i <= Math.sqrt(N); i++) {
        if (N % i == 0) {
            count += 2;
        }
        if (i == Math.sqrt(N))
            count--; // avoid over-counting
    }
    return count;
}
```
Question 18

• Mid-level question
• The requirement: compare two strings
• How do you compare two strings?
  – If they are not in the same length, they can not be equal
  – If they are in the same length, then if there is one different pair of characters, they can not be equal
  – If we can not find anything wrong, they are equal
  – However, if we find a pair of matching characters, it does not mean that they are equal
Question 18

• If they are not in the same length, they aren’t equal

  \[
  \begin{array}{cccc}
  H & o & w & a & r & e \\
  H & o & w & a & r & e & y & o & u & ?
  \end{array}
  \]

• If they are in the same length, then if there is one different pair of characters, they can not be equal

  \[
  \begin{array}{cccc}
  H & o & w & a & r & e & y & o & u & ? \\
  H & o & w & i & s & g & o & i & n & g
  \end{array}
  \]

• However, if we find a pair of matching characters, it does not mean that they are equal

  \[
  \begin{array}{cccc}
  H & o & w & a & r & e & y & o & u & ? \\
  H & o & w & i & s & g & o & i & n & g
  \end{array}
  \]
public boolean equalStrings(String a, String b) {
    boolean result = true;
    // We start from true, and try to find violations
    if (a.length() != b.length()) {
        result = false;
    } else {
        for (int i = 0; i < a.length(); i++) {
            if (a.charAt(i) != b.charAt(i))
                result = false;
            // You can not write: else result = true;
        }
    }
    return result;
}
Question 18

• Another idea
  – Count the pairs and see if there are a.length() pairs

```java
public boolean equalStrings(String a, String b) {
    int match = 0;
    if (a.length() != b.length()) {
        return false;
    } else {
        for (int i = 0; i < a.length(); i++) {
            if (a.charAt(i) == b.charAt(i))
                match++;
        }
    }
    return (match == a.length());
}
```
Question 18

• Advanced version
  – This is the most difficult question in the exam. It’s good that almost no one attempted it.

• Basic idea
  – Find the short string, and compare each character in the short string with the long string
  – After that, check if all remaining characters in the long strings are all spaces
Question 19

• Mid-level question (it is not a hard question)
• The requirement: calculate \( \pi \) using the series

\[
\frac{\pi}{4} = \frac{1}{1} - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \frac{1}{13} - \frac{1}{15} \ldots
\]

• How to calculate?
  – Of course you have to use a loop
  – But what is in the loop body?
  – **The key point is to find the pattern**
Question 19

\[ \frac{\pi}{4} = \frac{1}{1} - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \frac{1}{13} - \frac{1}{15} \cdots \]

• Idea 1: (Find the relationship in terms and indices)
  – The 1\textsuperscript{st} term is \(1/(2*1-1)\), it is positive
  – The 2\textsuperscript{nd} term is \(1/(2*2-1)\), it is negative
  – ...
  – The i\textsuperscript{th} term is \(1/(2*i-1)\), it is positive if i is odd, and is negative if i is even
public double pi() {
    double qPi = 0;
    for (int i = 1; i <= 10000; i++) {
        if (i % 2 != 0) { // check pos or neg
            qPi += 1 / (double) (2 * i - 1);
            // don't forget the type converting!
        } else {
            qPi -= 1 / (double) (2 * i - 1);
        }
    }
    return qPi * 4;
    // remember: we are calculating a quarter of pi
}
Question 19

\[
\frac{\pi}{4} = \frac{1}{1} - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \frac{1}{13} - \frac{1}{15} \ldots
\]

• Idea 2: (Find the relationship in terms and indices)
  – The 1\textsuperscript{st} term is 1/1
  – The 2\textsuperscript{nd} term is the 1\textsuperscript{st} term multiplying -1/3
  – The 3\textsuperscript{rd} term is the 2\textsuperscript{nd} term multiplying -3/5
  – ...
  – The i\textsuperscript{th} term is the (i-1)\textsuperscript{th} term multiplying -(2*i-1)/(2*i+1)
public double pi () {
    double pi = 0;
    double term = 4;
    for (int i = 1; i <= 10000; i++) {
        pi += term;
        term *= -(double) (2 * i - 1) / (2 * i + 1);
    }
}
Question 19

\[
\frac{\pi}{4} = \frac{1}{1} - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \frac{1}{13} - \frac{1}{15} \ldots
\]

• Idea 3: (Find the relationship in only terms)
  – The 1\textsuperscript{st} term’s divisor is 1
  – The 2\textsuperscript{nd} term’s divisor is 3
  – ...
  – The i\textsuperscript{th} term’s divisor is the (i-1)\textsuperscript{th} term’s divisor plus 2
  – The positive and negative term alters
  – End the loop when the divisor is greater than 20001
public double pi() {
    double qPi = 0, divisor = 1;
    boolean odd = true;
    while (divisor <= 20001) { // Notice the condition
        if (odd) {
            qPi += (1 / divisor);
            odd = false;
            // if current term is pos, turn to neg;
        } else {
            qPi -= (1 / divisor);
            odd = true;
            // if current term is neg, turn to pos;
        }
        divisor += 2;
    }
    return qPi * 4;
}
Question 19

\[
\frac{\pi}{4} = \frac{1}{1} - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \frac{1}{13} - \frac{1}{15} \ldots
\]

• Idea 4: (Find the relationship in only terms)
  – The 1\textsuperscript{st} two terms are 1/1-1/3
  – The 2\textsuperscript{nd} two terms are 1/5-1/7
  – ......
  – The i\textsuperscript{th} two terms’ divisors are the (i-1)\textsuperscript{th} two terms’ divisors plus 4
  – You can group the terms by pairs – one by one is not a must
public double pi() {
    double qPi = 0;
    double divisor = 1;
    for (int i = 0; i < 10000; i++) {
        qPi += (1 / divisor);
        qPi -= (1 / (divisor + 2));
        divisor += 4;
    }
    return qPi * 4;
}
Let Me Know If

- I added the numbers wrong on your exam paper
  - I won’t re-grade your answers
- The grade on the paper is different from Sakai
- You attempted Question 19 and want to see the correct code in your version
  - All semi-finished answers in Question 19 were modified to complete programs when being graded
- You have been working hard but feels that the grade can not reflect your effort