CIS 1068

November 14, 2017
Administrative Stuff

- Netflix Challenge
- New assignment posted soon
- Lab grades
Last Time. Building Our Own Classes

- Why
- Abstraction
- More on the `new` operator
- Fields
- Class vs the user of the class (aka client)
public class WhatsPrinted2 {
    public static void main(String args[]) {
        String s1="";
        for (int i=0; i<2; i++) {
            String s2 = "baa";
            for (int j=0; j<2; j++) {
                if ((i+j)%2==0) {
                    s1+="a";
                } else {
                    s2+=s2;
                }
            }
        }
        System.out.println(s2);
    }
}
public class WhatsPrinted2 {
    public static void main(String args[]) {
        String s1="";
        for (int i=0; i<2; i++) {
            String s2 = "baa";
            for (int j=0; j<2; j++) {
                if ((i+j)%2==0) {
                    s1+="a"
                } else {
                    s2+=s2;
                }
            }
        }
        System.out.println(s2);
    }
}

Answer
s2 is out of scope. Error
How much is a megabyte?
How much is a megabyte? about a million or $2^{20}$
public class Mystery11 {
    public static void func(int x, int y, int z) {
        if (x>y+z) {
            x+=y;
        }
        y--;
        z=z%2;
    }
    public static void main(String args[]) {
        int x=10, y=20, z=30;
        func(z, x, y);
        System.out.println(x);
    }
}
public class Mystery11 {
    
    public static void func(int x, int y, int z) {
        if (x>y+z) {
            x+=y;
        }
        y--;
        z=z%2;
    }
    
    public static void main(String args[]) {
        int x=10, y=20, z=30;
        func(z, x, y);
        System.out.println(x);
    }
}

answer

x is 10. func( ) operates on copies, not the originals
import java.util.Arrays;

public class Mystery07 {
    public static final int SIZE=10;

    public static void update(int A[], int x) {
        for (int i=0; i<A.length; i++) {
            A[i] += x;
        }
    }

    public static void main(String args[]) {
        int A[] = new int[SIZE];
        for (int i=0; i<A.length; i++) {
            A[i] = i*10;
        }
        update(A, 6);
        System.out.println(A[1]);
    }
}
```java
import java.util.Arrays;

public class Mystery07 {
    public static final int SIZE=10;
    public static void update(int A[], int x) {
        for (int i=0; i<A.length; i++) {
            A[i]+=x;
        }
    }

    public static void main(String args[]) {
        int A[] = new int[SIZE];
        for (int i=0; i<A.length; i++) {
            A[i]=i*10;
        }
        update(A, 6);
        System.out.println(A[1]);
    }
}
```

Answer
16
public class Mystery12 {
    public static void func(int x, int y, int z) {
        if (x>y+z) {
            x+=y;
        }
        y--;
        z=z%2;
        System.out.println(x);
    }
    public static void main(String args[]) {
        int x=11, y=19, z=31;
        func(z, x, y);
    }
}
public class Mystery12 {
    public static void func(int x, int y, int z) {
        if (x>y+z) {
            x+=y;
        }
        y--;
        z=z%2;
        System.out.println(x);
    }
    public static void main(String args[]) {
        int x=11, y=19, z=31;
        func(z, x, y);
    }
}

Answer
42
public class WhatsPrinted5 {
    public static int func(int y) {
        return y*2;
    }

    public static void main(String args[]) {
        int x=10;
        func(x);
        System.out.println(x);
    }
}
public class WhatsPrinted5 {
    public static int func(int y) {
        return y*2;
    }

    public static void main(String args[]) {
        int x=10;

        func(x);
        System.out.println(x);
    }
}

Answer
10. Again, `func()` operates on copies, not the original. Return value not read.
public class WhatsPrinted3 {
    public static void whatsPrinted(int A[]) {
        for (int i=1; i<A.length; i++) {
        }
    }
    public static void main(String args[]) {
        int A[] = {9,2,6,8,7};
        whatsPrinted(A);
        System.out.println(A[A.length-1]);
    }
}
public class WhatsPrinted3 {
    public static void whatsPrinted(int A[]) {
        for (int i=1; i<A.length; i++) {
        }
    }

    public static void main(String args[]) {
        int A[] = {9,2,6,8,7};
        whatsPrinted(A);
        System.out.println(A[A.length-1]);
    }
}

Answer
36
Suppose that we have A, which is a two-dimensional array of int. Which of the following gives us the item in A which is stored in the first row, last column?

- A[length-1][0]
- A[0][A[0].length-1]
- Impossible to tell with the information given.
- A[0][length-1]
- A[0].length-1
Suppose that we have A, which is a two-dimensional array of int. Which of the following gives us the item in A which is stored in the first row, last column?

- \( A[\text{length}-1][0] \)
- **\( A[0][A[0].\text{length}-1] \)**
- Impossible to tell with the information given.
- \( A[0][\text{length}-1] \)
- \( A[0].\text{length}-1 \)
Which analogy is most accurate?

- cookie cutter is to cookie as object is to class
- cookie is to cookie cutter as object is to class
- cookie cutter is to cookie as blueprint is to object
- cookie is to cookie cutter as blueprint-class is to class
- cookie is to cookie cutter as base class is to object
Which analogy is most accurate?

- cookie cutter is to cookie as object is to class
- **cookie is to cookie cutter as object is to class**
- cookie cutter is to cookie as blueprint is to object
- cookie is to cookie cutter as blueprint-class is to class
- cookie is to cookie cutter as base class is to object
public class WhatsPrinted1 {
    public static void main(String args[]) {
        int x = 0;
        do {
            System.out.print(x + " ");
            x--;
        } while (x > 10);
        System.out.println();
    }
}
public class WhatsPrinted1 {
    public static void main(String args[]) {
        int x=0;
        do {
            System.out.print(x + " ");
            x--;
        } while (x>10);
        System.out.println();
    }
}

Answer
0. With a do-while loop, we always do the body at least once.
public class MysteryShack {
    public static int[] mystery(int A[]) {
        int B[] = new int[A.length];

        for (int i=0; i<A.length; i++) {
        }

        return B;
    }

    public static void main(String args[]) {
        int A[] = {100, 200, 300, 400, 500};
        int B[] = {10, 20, 30, 40, 50};
        int C[];

        C = mystery(B);
        if (C == A) {
            System.out.println("same");
        } else {
            System.out.println("different");
        }
    }
}
public class MysteryShack {
    public static int[] mystery(int A[]) {
        int B[] = new int[A.length];
        for (int i=0; i<A.length; i++) {
            B[i]=A[i]*10;
        }
        return B;
    }
    public static void main(String args[]) {
        int A[]={100,200,300,400,500};
        int B[]={10,20,30,40,50};
        int C[];
        C=mystery(B);
        if (C==A) {
            System.out.println("same");
        } else {
            System.out.println("different");
        }
    }
}

Answer
different. == on references compares locations of what’s being referenced, not the content of what’s being referenced
public class Mystery08 {
    public static final int SIZE=10;
    public static void func(int A[], int B[], int x) {
        for (int i=0; i<B.length; i++) {
        }
        A=B;
    }
    public static void main(String args[]) {
        int []A = new int[SIZE];
        int []B = new int[SIZE];

        for (int i=0; i<A.length; i++) {
            A[i]=i;
            B[i]=i*10;
        }
        func(A, B, 7);
        System.out.println(A[1]);
    }
}
public class Mystery08 {
    public static final int SIZE=10;
    public static void func(int A[], int B[], int x) {
        for (int i=0; i<B.length; i++) {
        }
        A=B;
    }
    public static void main(String args[]) {
        int []A = new int[SIZE];
        int []B = new int[SIZE];
        for (int i=0; i<A.length; i++) {
            A[i]=i;
            B[i]=i*10;
        }
        func(A, B, 7);
        System.out.println(A[1]);
    }
}

Answer
1. The reference in func is copied (A). This doesn’t affect the A in main.
Write the few lines of code that prints all of the integers from 500 to 5000 that are evenly divisible by 7 using a for loop, while loop, and do-while loop.
Write the few lines of code that prints all of the integers from 500 to 5000 that are evenly divisible by 7 using a for loop, while loop, and do-while loop.

```java
for (int i=500; i<=5000; i++) {
    if (i%7==0)
        System.out.println(i);
}

int i=500;
while (i<=5000) {
    if (i%7==0)
        System.out.println(i);
    i++;
}

int i=500;
do {
    if (i%7==0)
        System.out.println(i);
    i++;
} while (i<=5000);
```
Write the single line of code which declares a two-dimensional array of double with NCOL columns and NROW rows.

```java
double A[][]=new double[NROW][NCOL];
or
double [][]A=new double[NROW][NCOL];
```
Write the single line of code which declares a two-dimensional array of double with NCOL columns and NROW rows.

**Answer**

double A[][]=new double[NROW][NCOL];

or

double [][]A=new double[NROW][NCOL];
Write a method which is passed an array of int[ ] A and an int t (for \textit{threshold}). The method returns the number of odd items in A which are greater than or equal to t.

Write a method which is passed a two dimensional array of int A[ ][], and an int t. The method returns the index of the row in A which contains the greatest number of odd numbers greater than or equal to t. The method returns -1 if no row has any odd numbers greater than \( t \). You may use your answer to part a in your answer to this part.
public static int numOddGreater(int A[], int t) {
    int numOdd=0;

    for (int i=0; i<A.length; i++) {
        if (A[i]%2==1 && A[i]>=t)
            numOdd++;
    }

    return numOdd;
}
public static int mostOddGreater(int A[][], int t) {
    int rowMostOdd = 0;
    int mostOdd = numOddGreater(A[0], t);

    for (int i = 1; i < A.length; i++) {
        int currentRow = numOddGreater(A[i], t);
        if (currentRow > mostOdd) {
            mostOdd = currentRow;
            rowMostOdd = i;
        }
    }

    if (mostOdd == 0)
        return -1;

    return rowMostOdd;
}
Write a method which is passed the name of a file. The method returns the sum of the lengths of the words in the file that begin with a vowel. You may assume that the file contains nothing but words and inter-word spacing (i.e., it contains no punctuation, numbers, etc.). You are not responsible for handling FileNotFoundException.
public static int lenStartsWithVowels(String filename) throws FileNotFoundException {
    int sum=0;
    Scanner in = new Scanner(new File(filename));

    while (in.hasNext()) {
        String word = in.next();
        if (isVowel(word.charAt(0))) {
            sum+=word.length();
        }
    }
    in.close();
    return sum;
}

public static boolean isVowel(char c) {
    String vowels="aeiouAEIOU";
    for (int i=0; i<vowels.length(); i++) {
        if (c==vowels.charAt(i)) {
            return true;
        }
    }
    return false;
}
Write the few lines of code (you do not need a complete class, main( ), import statements, etc.), that prompts the user at the keyboard to enter a series of temperatures followed by a sentinel value of -100 to quit. The program then prints the average of the cool temperatures (i.e., those that were below 40 degrees) or the word ”balmy” if all of the temperatures were 40 or above.
```java
final int SENTINEL=-100;
final int COLD=40;

int numColdTemps=0;
int sumColdTemps=0;

Scanner in = new Scanner(System.in);
System.out.print("Enter temp >");
int cur = in.nextInt();
while (cur != SENTINEL) {
    if (cur<COLD) {
        numColdTemps++;
        sumColdTemps+=cur;
    }
    System.out.print("Enter temp >");
    cur = in.nextInt();
}
if (numColdTemps>0) {
    System.out.println(((double)sumColdTemps)/numColdTemps);
} else {
    System.out.println("balmy");
}
```