CSE 21
Intro to Computing II

Lecture 4 – Methods (3)

Today

- Methods (3)
- Lab
  - Lab 3 due this week (2/11 – 2/17)
  - Lab 4 assigned this week
    - Generic cheese shop v2
  - Due in one week
    - Make sure to show your work to YOUR OWN TA (or me) before submission
      - Not required but highly encouraged to make sure you receive full credit
- Reading Assignment
  - Sections 7.1 – 7.4 (including participation activities)
    - Work on the Participation Activities in each section to receive participation grade at the end of semester (based on at least 80% completion)

Next Week (2/18 – 2/24)

- No lecture (Pres. Day Holiday on Monday, 2/19)
- Lab 5 assigned (2/18 – 2/24)
  - Methods Overloading
  - Due the week after (2/25 – 3/3)
    - Make sure to show your work to YOUR OWN TA (or me) before submission
      - Not required but highly encouraged to make sure you receive full credit
- Project 1 Assigned
  - Posted on Friday (2/23)
  - Due in two weeks after (3/9)
  - Can work in pairs (each student must submit and indicate team member)

Extra Credit (reminder)

- Up to 5 percentage points of total grade
- Based on completion of challenge activities of reading assignment sections
  - 20% complete = 1% of total grade
  - 40% complete = 2% of total grade
  - 60% complete = 3% of total grade
  - 80% complete = 4% of total grade
  - 100% complete = 5% of total grade
- Like participation activity, scores evaluated at the end of semester

Sum Example: Scope (Review)

Output: Main num1 is 18
Main num2 is 13
First tally is 13
Second tally is is 18
Sum is 31

```
public class PreferenceMOSv2{
    // Method Declaration like variables (callee)
    public static int CombinedTally(int num1, int num2) {                    #6
        System.out.println("First tally is " + num1);                       #7
        System.out.println("Second tally is " + num2);                      #8
        int total = num1 + num2;                                            #9
        num1 = 100; ←————————————— No Effect: Logical Error                 #10
        return total;                                                       #11
    }

    public static void main(String[] args) {                                #1
        int num1 = 18, num2 = 13;                                           #2
        System.out.println("Main num1 is " + num1);                         #3
        System.out.println("Main num2 is " + num2);                         #4
        int total #12 = CombinedTally(num2, num1); // arguments switched    #5
        System.out.println("Sum is " + total);                              #13
    }
}
```

Two sets of variables: ***num1***, ***num2*** and ***total*** local to each method are completely independent!

Multiple Returns (Review)

tally[0] = 20
tally[1] = 20

```
public static int maxAndroidIOS(int num1, int num2) {          #3
    if (num1 > num2)                                            #4
        return num1;
    if (num2 > num1)                                            #5
        return num2;
    if (num2 == num1)                                           #6
        return num2;                                            #7
    return 0;
}

public static void main(String[] args) {                        #1
    …
    int maxNumber #8 = maxAndroidIOS(tally[0], tally[1]);       #2
    System.out.println("Max is " + maxNumber);                  #9
}
```

Multiple Returns Optimized (Review)

tally[0] = 5
tally[1] = 10

```
public static int maxAndroidIOSv1(int num1, int num2) {                    #3
        if (num1 > num2)                                                   #4
                return num1;
         return num2;                                                      #5
}

public static void main(String[] args) {                                   #1
        …
        int maxNumber #6 = maxAndroidIOSv1(tally[0], tally[1]);            #2
        System.out.println("Max is " + maxNumber);                         #7
}
```

Multiple Returns: Conditional (Review)

tally[0] = 5
tally[1] = 10

```
public static int maxAndroidIOSv2(int num1, int num2) {          #3
    return num1 > num2 ? num1:num2;                              #4
}
```

True False

```
public static void main(String[] args) {                         #1
    …
    int maxNumber #5 = maxAndroidIOSv2(tally[0], tally[1]);      #2
    System.out.println("Max is " + maxNumber);                   #6
}
```

CheeseShop.java

```
public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.println("We sell 3 kinds of  Cheese");
        System.out.println("Dalaran Sharp: $1.25 per pound");
        System.out.println("Stormwind Brie: $10.00 per pound");
        System.out.println("Alterac Swiss: $40.00 per pound");

        System.out.print("Enter the amount of Sharp: ");
        int sharp = input.nextInt();
        System.out.print("Enter the amount of Brie: ");
        int brie = input.nextInt();
        System.out.print("Enter the amount of Swiss:  ");
        int swiss = input.nextInt();

        double total = sharp*1.25 + brie*10.0 + swiss*40.00;

        double discount = 0;
                if (total >= 100)
                        discount = 25;
                else if (total >= 50)
                        discount = 10;
```

```
        System.out.print("Display the itemized list? (1 for yes) ");
        int itemized = input.nextInt();
        if (itemized == 1) {
                if (sharp > 0)
                        System.out.println(sharp + " lbs of Sharp
                        @ $1.25 = $" + sharp*1.25);
                if (brie > 0)
                        System.out.println(brie + " lbs of Brie @
                        $10.00 = $" + brie*10.0);
                if (swiss > 0)
                        System.out.println(swiss + " lbs of Swiss
                        @ $40.00 = $" + swiss*40.00);
        }

        System.out.println();
        System.out.println("Sub Total: $" + total);
        System.out.println("-Discount: $" + discount);
        System.out.println("Total    : $" + (total - discount));
}
```

Cheese Shop (6 processes)

A. List all the cheese types available and the prices
B. Asks the user how many pounds of each type of cheese to purchase
C. Calculate Sub Total (price*amount of each cheese added together)
D. Discount of Sub Total
   - A $10 discount if their purchase is $50 or over
   - An additional $15 discount ($25 total) if $100 or over
E. Ask the user if they would like to see a list of what they purchased
   - If yes, a list comes up showing how much of each type of cheese they bought and the cost of each cheese
   - Display only the cheese they actually bought
   - If user answered no, then no itemized information is displayed
F. Display Sub Total, Discount and Total Price

CheeseShop.java

```
public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.println("We sell 3 kinds of  Cheese");
        System.out.println("Dalaran Sharp: $1.25 per pound");
        System.out.println("Stormwind Brie: $10.00 per pound");
        System.out.println("Alterac Swiss: $40.00 per pound");
```

A

```
        System.out.print("Enter the amount of Sharp: ");
        int sharp = input.nextInt();
        System.out.print("Enter the amount of Brie: ");
        int brie = input.nextInt();
        System.out.print("Enter the amount of Swiss:  ");
        int swiss = input.nextInt();
```

B

```
        double total = sharp*1.25 + brie*10.0 + swiss*40.00;
```

C

```
        double discount = 0;
        if (total >= 100)
                discount = 25;
        else if (total >= 50)
                discount = 10;
```

D

```
        System.out.print("Display the itemized list? (1 for yes) ");
        int itemized = input.nextInt();
        if (itemized == 1) {
                if (sharp > 0)
                        System.out.println(sharp + " lbs of Sharp
                        @ $1.25 = $" + sharp*1.25);
                if (brie > 0)
                        System.out.println(brie + " lbs of Brie @
                        $10.00 = $" + brie*10.0);
                if (swiss > 0)
                        System.out.println(swiss + " lbs of Swiss
                        @ $40.00 = $" + swiss*40.00);
        }
```

E

```
        System.out.println();
        System.out.println("Sub Total: $" + total);
        System.out.println("-Discount: $" + discount);
        System.out.println("Total    : $" + (total - discount));
}
```

F

CheeseShopV2.java

```
public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

         intro(); A

        int sharp = getAmount(input, "Sharp"); B
        int brie = getAmount(input, "Brie");
        int swiss = getAmount(input, "Swiss");

        double total = calcSubTotal(sharp, brie, swiss); C

        System.out.print("Display the itemized list? (1 for yes) ");
        int itemized = input.nextInt();
        if (itemized == 1)
                itemizedList(sharp, brie, swiss); E

        System.out.println();
        printTotal(total, discount(total));
}
```

Methods

F D

How to calculate discount?

- $10 discount if total purchase is $50 or over and an **additional** $15 discount ($25 total) if total purchase is $100 or over:
  - if >= $50 then -$10 AND if >= $100 then extra -$15
  - if >= $100 then -$25 OR if >= $50 then -$10
  - if >= $50 then -$10 OR if >= $100 then -$25

Break it down into simple logical steps!

How to Translate to code?

- If >= $50 then -$10 AND if >= $100 then extra -$15

```
int discount = 0;
if (total >= 50)
    discount -= 10;
if (total >= 100)
    discount  -= 15;
```

- If >= $100 then -$25 OR if >= $50 then -$10

```
int discount = 0;
if (total >= 100)
    discount = -25;
else if (total >= 50)
    discount  = -10;
```

- If >= $50 then -$10 OR if >= $100 then -$25

```
int discount = 0;
if (total >= 50)
    if (total >= 100)
        discount  = -25;
    else
        discount = -10;
```

Return styles

- If >= $50 then -$10 AND if >= $100 then extra -$15

```
int discount = 0;
if (total >= 50)
    discount -= 10;
if (total >= 100)
    discount  -= 15;
return discount;
```

- If >= $100 then -$25 OR if >= $50 then -$10

```
if (total >= 100)
    return  -25;
else if (total >= 50)
    return  -10;
return 0;
```

- If >= $50 then -$10 OR if >= $100 then -$25

```
int discount = 0;
if (total >= 50)
    if (total >= 100)
        return discount  = -25;
    else
        return discount = -10;
return 0;
```

What is "discount"?

```
public static int discount(double subTotal){
	int discount = 0;
	if (subTotal >= 100)
		discount = -25;
	else if (subTotal >= 50)
		discount = -10;
	return discount;
}

public static void main(String[] args) {
	double total = …
	int discount = discount(total);
	printTotal(total, discount);
```

How many discounts are there?

What is “discount”?

```
public static int discount(double subTotal){
        int discount = 0;
        if (subTotal >= 100)
                discount = 25;
        else if (subTotal >= 50)
                discount = 10;
        return discount;
}

public static void main(String[] args) {
        double total = …
        int discount = discount(total);
        printTotal(total, discount);
```

1) Method , has ()

What is “discount”?

```
public static int discount(double subTotal){
    int discount = 0;
    if (subTotal >= 100)
        discount = 25;
    else if (subTotal >= 50)
        discount = 10;
    return discount;
}

public static void main(String[] args) {
    double total = …
    int discount = discount(total);
    printTotal(total, discount);
```

2) Local variable in method

What is “discount”?

```
public static int discount(double subTotal){
    int discount = 0;
    if (subTotal >= 100)
            discount = 25;
    else if (subTotal >= 50)
            discount = 10;
    return discount;
}
```

3) Local variable in main

```
public static void main(String[] args) {
    double total = …
    int discount = discount(total);
    printTotal(total, discount);
```

Name Overloading

- Name resolution is Scope dependent
  - Names of objects have their own scope
    - A variable declared in a for-loop does not exist outside the loop
    - A variable declared in a method does not exist outside the method
- Variables just use the name
- Methods are declared and invoked using parentheses **( )**
- Both require types
  - int x; public static float getFloat();

Method overloading

Are we allowed to have multiple methods of the same name???

```
public static int getAmount(Scanner input, String name) { // 1
        System.out.print("Enter the amount of " + name + ": ");
        int amount = input.nextInt();
        return amount;
}
```

2 input parameters: Scanner + String

```
public static void getAmount(Scanner input, String[] names, int[] amounts) { // 2
        for (int i = 0; i < names.length; i++) {
                System.out.print("Enter the amount of " + names[i] + " : ");
                amounts[i] = input.nextInt();
        }
}
```

3 input parameters: Scanner + String pointer + int pointer

```
public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        String[] names = new String[3]; int[3] amounts = new int[3];
        int sharp = getAmount(input, "Sharp");
        int brie = getAmount(input, "Brie");
        int swiss = getAmount(input, "Swiss");
        getAmount(input, names, amounts);
```

2 arguments: Scanner + String

3 arguments: Scanner + String[] + int[]

Type of arguments determines the method call!

Matching method calls

For the parameters in the method declarations below

- Assume **input** is of type Scanner, **names** is of type String[], **amounts** is of type int[]

- getAmount(input, "Random");
  - Scanner + String // Match 1
- getAmount("Random", input);
  - String + Scanner // Don't match
- getAmount(input, names[0]);
  - Scanner + String // Match 1
- getAmount(input, names);
  - Scanner + String[] // Don't match
- getAmount(input, names, amounts);
  - Scanner + String[] + int[] // Match 2
- getAmount(input, amounts, names);
  - Scanner + int[] + String[] // Don't match
- getAmount(input, names[0], amounts[0]);
  - Scanner + String + int // Don't match
- getAmount(input, names, new int[MAXCHEESE]
  - Scanner + String[] + int[] // Match 2
- getAmount(input, names, prices);
  - Scanner + String[] + double[] // Don't match

```
public static int getAmount(Scanner input, String name) // 1
public static void getAmount(Scanner input, String[] names, int[] amounts) // 2
```

Sum All (a review)

- Summation of numbers 1 to max
  - Steps
    - subTotal = 0;
    - subTotal += 1;
    - subTotal += 2;
    - ….
    - subTotal += max;
  - Loop
    - Begin: 1
    - End: max
    - Increment: increase by 1
    - Body: add current number to running total

For-loop Forms

```
for (int i = 1; i <= max ; i++) {
    subTotal += i;
}
```

i = 1, 2, 3, …, max (#iterations = max)

```
for (int i = 0; i < max; i++) {
    subTotal += i + 1;
}
```

i = 0, 1, 2, …, max-1 (#iterations = max)

```
for (int i = max; i > 0; i--) {
    subTotal += i;
}
```

i = max, max-1, max-2, …, 1 (#iterations = max)

Be aware of how many iterations the loop runs!

SumAll Method

```
public static int sumAll(int max) {
    int subTotal = 0;
    for (int i = 1; i <= max ; i++) {
        subTotal += i;
        System.out.println("sumAll " +  i + " value " + subTotal);
    }
    return subTotal;
}
```

```
in main ....
    sumAll(5);
    sumAll(10);
    sumAll(20);
    sumAll(15);
```

Run Result

```
sumAll 1 value 1
sumAll 2 value 3
sumAll 3 value 6
sumAll 4 value 10
sumAll 5 value 15
sumAll output for 5 is 15

sumAll 1 value 1
sumAll 2 value 3
sumAll 3 value 6
sumAll 4 value 10
sumAll 5 value 15
sumAll 6 value 21
sumAll 7 value 28
sumAll 8 value 36
sumAll 9 value 45
sumAll 10 value 55
sumAll output for 10 is 55
```

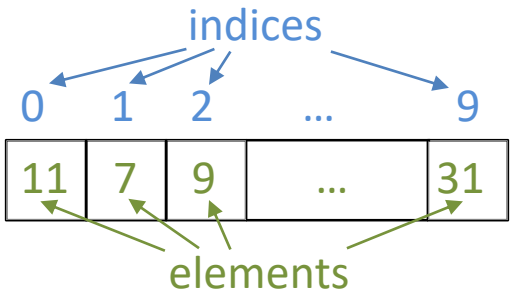
```
sumAll 1 value 1
sumAll 2 value 3
sumAll 3 value 6
sumAll 4 value 10
sumAll 5 value 15
sumAll 6 value 21
sumAll 7 value 28
sumAll 8 value 36
sumAll 9 value 45
sumAll 10 value 55
sumAll 11 value 66
sumAll 12 value 78
sumAll 13 value 91
sumAll 14 value 105
sumAll 15 value 120
sumAll 16 value 136
sumAll 17 value 153
sumAll 18 value 171
sumAll 19 value 190
sumAll 20 value 210
sumAll output for 20 is 210
```

```
sumAll 1 value 1
sumAll 2 value 3
sumAll 3 value 6
sumAll 4 value 10
sumAll 5 value 15
sumAll 6 value 21
sumAll 7 value 28
sumAll 8 value 36
sumAll 9 value 45
sumAll 10 value 55
sumAll 11 value 66
sumAll 12 value 78
sumAll 13 value 91
sumAll 14 value 105
sumAll 15 value 120
sumAll output for 15 is 120
```

Understanding Arrays

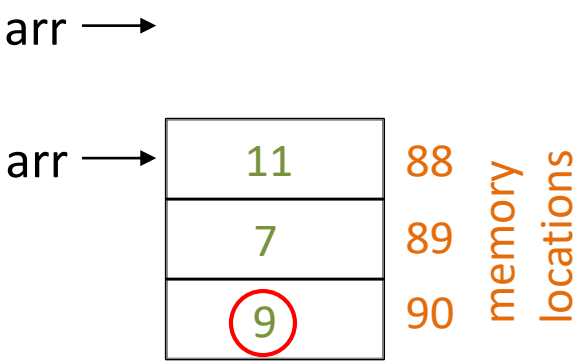
- One variable storing a list of data items

```
int[] arr = {11, 7, 9, 4, 55, 2, 1, 18, 2, 31};
```



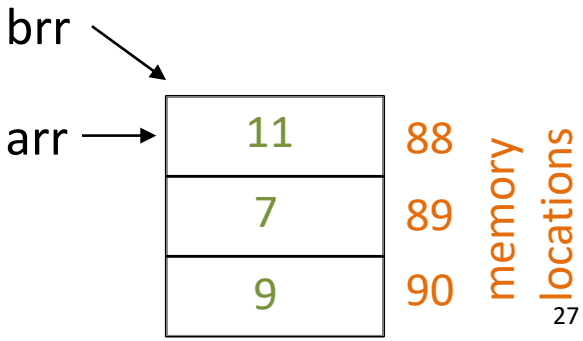
- Another view of arrays
  - An array variable is a *reference variable*
    - A pointer to a memory location

```
int[] arr;

arr = new int[3];
arr[0] = 11; arr[1] = 7; arr[2] = 9;
```



    - Internally arr "stores" the memory location 88
    - When we write arr[2], internally we retrieve the element stored at memory location 88 + 2 (in this case, 9)
    - How about two variables pointing to the same array?

```
int[] brr;
brr = arr; // brr now "stores 88" as well
```



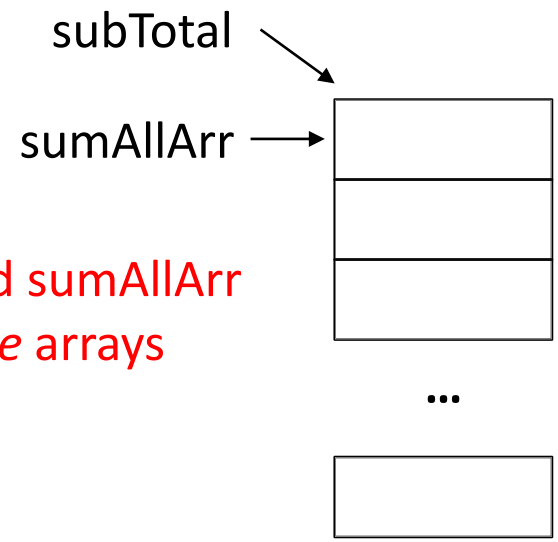
Array of subTotals

```
public static int sumAll(int[] subTotal, int max) {

    for (int i = 1; i <= max ; i++) {
        if (subTotal[i] == 0) {            // Empty slot…calculate
            subTotal[i] = subTotal[i-1] + i;
            System.out.println("sumAll[" + i + "] value is " + subTotal[i]);
        }
    }

    return subTotal[max];
}
```

1	2	3	4	5	6	7	8	9	10
1	3	6	10	15	21	28	36	45	55

Array parameter in Methods

```
public static void sumAll(int[] subTotal, int max) {
        for (int i = 1; i <= max; i++)
                if(subTotal[i] == 0)
                        subTotal[i] = subTotal[i-1] + i;
}

public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        int[] sumAllArr = new int[1000];
        for (int i = 0; i < 1000; i++) sumAllArr[i] = 0;
        int repeat = 0;
        do {
                System.out.print("Enter the max number for sumAll (between 0 and 1000): ");
                int max = input.nextInt();
                sumAll(sumAllArr, max);
                for (int i = 0; i <= max; i++)
                        System.out.println("sumAllArr[" + i + "] value is " + sumAllArr[i]);
                System.out.print("Repeat this program? (1 for yes) ");
                repeat = input.nextInt();
        } while (repeat == 1);
}
```



subTotal and sumAllArr are the *same* arrays

Run Result

In main() ….

```
sumAll(sumAllArr, 5);
sumAll(sumAllArr, 10);
sumAll(sumAllArr, 20);
sumAll(sumAllArr, 15);
```

```
sumAllArr[1] value is 1
sumAllArr[2] value is 3
sumAllArr[3] value is 6
sumAllArr[4] value is 10
sumAllArr[5] value is 15
sumAll output for 5 is 15
sumAllArr[6] value is 21
sumAllArr[7] value is 28
sumAllArr[8] value is 36
sumAllArr[9] value is 45
sumAllArr[10] value is 55
sumAll output for 10 is 55
sumAllArr[11] value is 66
sumAllArr[12] value is 78
sumAllArr[13] value is 91
sumAllArr[14] value is 105
sumAllArr[15] value is 120
sumAllArr[16] value is 136
sumAllArr[17] value is 153
sumAllArr[18] value is 171
sumAllArr[19] value is 190
sumAllArr[20] value is 210
sumAll output for 20 is 210
sumAll output for 15 is 120
```