**Chapter 5 Test Review**

**Free Response # 1** - Write a method, *calculatePassingExamScore that receives a semester grade as a double and returns the minimum final exam score that will give that student a passing grade in the class (also as a double). If they are not able to pass the class, regardless of exam score, the method should return a value of -1.0; You can assume, like at EK, that the semester grade counts for 80% of the final grade and the exam counts for 20%.*

**Follow the Logic**

**int** a = *convertBtoI*(*willError*(52, 13), *isLucky*(702));

System.***out***.println(a);

**int** b = *flipIt*(*findDigit*(129), *reduceIt*(21, 6, 4));

System.***out***.println(b);

**int** c = *cubeIt*(*finalCount*(0, 24));

System.***out***.println(c);

**double** d = *adjustGPA*(*isEligible*(*areTwins*(3, -3), *sum*(10,20,25)), *toDouble*(*reduceIt*(35, 11, 3)));

System.***out***.println(d);

**public** **static** **double** toDouble(**int** num){

 **return** (**double**) num;

 }

**public** **static** **int** finalCount(**int** numAces, **int** handCount){

 **while**(handCount >21 && numAces>0)

 {

 handCount-=10;

 numAces--;

 }

 **return** handCount;

 }

**public** **static** **double** sum(**double** a, **double** b, **double** c){

 **return** a + b + c;

 }

**public** **static** **int** sum(**int** a, **int** b, **int** c){

 **return** a + b;

 }

**public** **static** **boolean** isLucky(**int** x){

 **while**(x!=0)

 {

 **if**(x%10==7) **return** **true**;

 x /=10;

 }

 **return** **false**;

 }

**public** **static** **boolean** isEligible(**boolean** isCitizen, **int** age){

 **return** isCitizen && age >=35;

 }

**public** **static** **boolean** willError(**int** x, **int** y){

 **return** x % y != 0 || y == 0;

 }

**public** **static** **boolean** areTwins(**int** x, **int** y){

 **return** x==y || x == y \*-1;

 }

**public** **static** **double** adjustGPA(**boolean** isAPClass, **double** classGPA){

 **if**(!isAPClass) **return** classGPA;

 **return** classGPA + 1;

 }

**public** **static** **int** findDigit(**int** x){

 **if**(x < 10) **return** -1;

 **return** x / 10 % 10;

}

**public** **static** **int** cubeIt (**int** x){

 **if** (x<20) **return** (**int**)(Math.*pow*(x, 3));

 **return** x/2;

}

**public** **static** **int** convertBtoI(**boolean** a, **boolean** b){

 **if**(a && b) **return** 10;

 **else** **if** (!a && !b) **return** 5;

 **return** 3;

}

**public** **static** **int** reduceIt(**int** x, **int** y, **int** z){

 **if**(x < z) **return** z;

 **else**

 {

 **while**(x>=z)

 {

 x -=y;

 }

 }

 **return** x;

}

**public** **static** **int** flipIt(**int** x, **int** y){

 **double** z = Math.*pow*(x, y);

 z = 1/z;

 **return** (**int**)(z\*100)%10;

}

**LiveLab**

**Do P\_GoodPrimes.** A “good prime” is a prime number whose square is more than the product of the prime number before and after it. The user enters a number, x, and the your program will output the first x “good primes. The program will also output at the end the difference between the last two good primes.

2, 3, 5, 7, 11, 13, 17, 19 – Of the first eight prime numbers, 5, 11 and 17 are “good primes”.

52 = 25 which is greater than 3 \* 7

Sample Run:

User input: 4 Outputs first 4 good primes and difference between the last two.

5 11 17 29 12