**Name: zhenzhen su**

**Class: CSC 143**

**Instructor: Dr. Charles Reid**

1. Problem 4 : write a recursive method called doubleDigits that accepts an integer n and returns the integer obtained by replacing every digit of n with two of that digit.

Code:

```
public class Main {
    public static void main(String[] args) {
        System.out.println("The double digits for 0 is " + doubleDigits(0));
        System.out.println("The double digits for 348 is " + doubleDigits(348));
        System.out.println("The double digits for -789 is " + doubleDigits(-789));
    }

            //precon: integer n
            //postcon: integer
            public static int doubleDigits(int n){
                if(n == 0){
                    return n;
                }else if(n > 0){
                    int count = 0;
                    int m = n;

                    while(m > 9){
                        count ++;
                         m = m/10;
                    }

                    int temp = (int)(n - m*pow(10,count));
                    m = (int)((m*10 + m)*pow(100,count));
                    return m + doubleDigits(temp);
                }else{
                    return -1 * doubleDigits(abs(n));
                }
            }
}
```

**Console output:**

```
The double digits for 0 is 0
The double digits for 348 is 334488
The double digits for -789 is -778899
```

2. **Problem 6: write a recursive method called writeSquares that accepts an integer as a parameter n and prints the first n squares separated by commas with odd squares in descending order followed by even squares in ascending order.**

**Code:**

```
public class Main {
    public static void main(String[] args) {
        writeSquares(9);
```

```
        writeSquares(1);
        writeSquares(0);
    }

       public static void writeSquares(int n){
           if(n < 1){
               throw new IllegalArgumentException();
           }else if(n == 1){
               System.out.print(1);
           }else{
               if(n%2 == 0){
                   writeSquares(n-1);
                   System.out.print(", " + n*n);
               }else{
                   System.out.print(n*n + ", ");
                   writeSquares(n-1);
               }
           }
       }

}
```

**Console output:**

```
81, 49, 25, 9, 1, 4, 16, 36, 64
1
Exception in thread "main" java.lang.IllegalArgumentException
        at com.company.Main.writeSquares(Main.java:49)
        at com.company.Main.main(Main.java:17)
```

3. **Problem 15: write recursive method called permut that accepts two integers n and r as parameters and returns the number of unique permutations of r items from a group of n items.**

Code:

```
public class Main {
    public static void main(String[] args) {
       System.out.println("The permutations for (6,3) is " + permut(6,3));
       System.out.println("The permutations for (7,4) is " + permut(7,4));

    }


       public static int permut(int n, int r){
           if(r == 1){
               return n;
           }else{
               return n * permut(n-1,r-1);
           }
       }
}
```

**Console output:**

The permutations for (6,3) is 120

The permutations for (7,4) is 840

4. Problem 22: write a recursive method called printSquares to find all ways to express an integer as a sum of squares of unique positive integers.

Code:

```
public class Main {
    public static void main(String[] args) {
        System.out.println("The sum of squares for 289 is below: ");
        printSquares(289);
        System.out.println("The sum of squares for 1 is below: ");
        printSquares(1);
        System.out.println("The sum of squares for 0 is below: ");
        printSquares(0);
        System.out.println("The sum of squares for 128 is below: ");
        printSquares(128);
   }



       public static void printSquares(int n ){
           explore( new ArrayList<Integer>(), n, 1);
       }

       private static void explore(ArrayList<Integer> list, int n, int num){
           if( n == 0){
               printHelper(list);

           }else if(n>0){
               for(int i= num; i<= Math.sqrt(n); i++){
                   if(n-i*i >= 0){
                       list.add(i);
                       explore(list,n-i*i,i+1);
                       list.remove(list.size()-1);
                   }
               }

           }
       }

       //Helper method to print sum of integers'square for printSquare method
       public static void printHelper(ArrayList<Integer> list){

           if(list.size() >= 1) {
               String result = list.get(0) + "^2";
               if (list.size() > 1) {
                   for (int i = 1; i < list.size(); i++) {
                       result = result + " + " + list.get(i) + "^2";
                   }
               }
               System.out.println(result);
           }
       }

}
```

**Console output:**

The sum of squares for 289 is below:

1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 7^2 + 8^2 + 11^2

1^2 + 2^2 + 3^2 + 5^2 + 9^2 + 13^2

1^2 + 3^2 + 4^2 + 5^2 + 6^2 + 9^2 + 11^2

1^2 + 3^2 + 5^2 + 6^2 + 7^2 + 13^2

2^2 + 3^2 + 4^2 + 8^2 + 14^2

2^2 + 3^2 + 5^2 + 7^2 + 9^2 + 11^2

2^2 + 4^2 + 5^2 + 6^2 + 8^2 + 12^2

2^2 + 4^2 + 5^2 + 10^2 + 12^2

2^2 + 4^2 + 6^2 + 8^2 + 13^2

2^2 + 4^2 + 10^2 + 13^2

2^2 + 5^2 + 8^2 + 14^2

2^2 + 8^2 + 10^2 + 11^2

3^2 + 6^2 + 10^2 + 12^2

8^2 + 9^2 + 12^2

8^2 + 15^2

17^2

The sum of squares for 1 is below:

1^2

The sum of squares for 0 is below:

The sum of squares for 128 is below: