Assignment 2

1. String class is immutable, which means that it is not possible to change or modify the contents once it is made. Methods in the String class will only construct and return a new string, and not modify it.

2.
```java
import java.util.Scanner;

class alpha{
    public static void main(String[] args){
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter a character: ");
        char input = scan.next().charAt(0);
        if( (input >= 'a' && input <= 'z')
            || (input >= 'A' && input <= 'Z')
            || (input >= '0' && input <= '9')){
            System.out.println("The character '" + input + "' is
            alphanumeric");
        } else {
            System.out.println("The character '" + input + "' is not
            alphanumeric");
        }
    }
}
```

3.
```java
import java.util.Scanner;

class whichCase{
    public static void main(String[] args){
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter a character: ");
        char input = scan.next().charAt(0);
        if( input >= 'a' && input <= 'z' ){
            System.out.println("The character '" + input + "' is
            lowercase");
        } else if( input >= 'A' && input <= 'Z' ) {
            System.out.println("The character '" + input + "' is
            uppercase");
        } else {
            System.out.println("The character '" + input + "' is
            neither lowercase or uppercase");
        }
    }
}
```

4. To convert hexadecimal to decimal, we must first make sure that if it is has a letter(s), it must be in uppercase, then we check how long the hexadecimal is. We will also need to prepare a variable that will store the decimal value of the hexadecimal that is initialized with 0, then for every character of the hexadecimal, as much as how long the hexadecimal is, the index will be based on the String "0123456789ABCDEF" where the index of the character will be added to 16 * (the decimal value). When all char of the hexadecimal has been used, that is the decimal of a hexadecimal.

5. i) Math.pow(2,2) will calculate for 2 to the power of 2 and will output a result of 4.
   ii) Math.max(2, Math.min(3,4)) will find the minimum of 3 and 4 which will result in 3, then will find the maximum of 2 and the minimum of 3 and 4 which is 3, hence will output a result of 3.
   iii) Math.round(2.5F) will round the number 2.5F to 3.
   iv) Math.ceil(-9.49) will round the number -9.49 up to -9.
   v) Math.floor(7.5) will round down the number 7.5 to 7.

6. i) contains is a method to check if a string contains a particular substring and returns a boolean value. For example, from String stuff = “BiNus University”; we then output the following, System.out.println(“Contains “University”: “ + stuff.contains("University")); which will output true, meaning that the string stuff contains “University”.
   ii) concat is a method to combine strings. For example, from String word1 = “BiNus” and String word2 = “University”, we can combine these strings to 1 string by doing String word3 = word1.concat(word2); and if we print word3, the output will be “BiNus University”.
   iii) compareTo is a method to compare a string with another string. If the 2 strings are equal, it will return 0, if it isn’t equal, it will return any other number besides 0. For example, String word1 = “Binus”; and String word2 = “Binus”; then we use the method in a selection, if((word1.compareTo(word2)) == 0) System.out.println(“Both words are equal to one another”); where the output will be “Both words are equal to one another” since both word1 and word2 is “Binus”.
   iv) format is a method that returns a formatted string. For example, if we have String kuliah = “BiNus University”; then we can also make String stuff = String.format(“%s %d”, kuliah, 1); we can just output it by doing System.out.println(stuff); and it will output “BiNus University 1.
   v) charAt is a method to point to an array in a String. For example, from String uni = “BiNus University”; when we use the method uni.charAt(4); and output it, the output will be the character in the string at the fourth index starting from 0 which is the character ‘s’.
   vi) replace is a method to overwrite all specific characters in a string into another specific character to the string. For example, from String stuff = “BiNus University”.replace(‘i’,’a’); where when we output the string it will result in “BaNus Unaversaty”.
   vii) substring is a method to take a part of a string based on the index chosen. For example, from String stuff = “BiNus University”; then we use stuff.substring(0,4); and when we output it, it will output “BiNus”.
   viii) trim is a method to eliminate all blank characters in a string. For example, from String stuff = “B i Nu s Uni versity”.trim(); it will eliminate all the blank characters in the string and when we output it afterward it will output “BiNusUniversity”.
   ix) toCharArray is to convert a string into an array of characters. For example, String stuff = “BiNus University”; we can then call on the method by doing the following, char[] word = stuff.toCharArray(); the string is then converted into an array of char. Then we can output it using a for loop as follows for(int i = 0;i<word.length();i++){ System.out.println(word[i]); } it will output “BiNus University”.
   x) split is a method to split a string from a regular expression and returns an array of characters. For example, String kuliah = “Bina Nusantara University” then we can use String[] words = kuliah.split(“\\s”); which will split the string based on whitespace and to output we will use a for loop as follows, for(String i: words){ System.out.println(i) }; it will output “Bina”,”Nusantara”,”University”.

xi) toLowerCase is a method that will make all characters of a string to lowercase. For example, from the String uni = “BiNus University”; when we use the method uni.toLowerCase(); and output it, the output will be “binus university”, notice that all the characters are lowercase characters.

xii) toUpperCase is a method that will make all characters of a string to uppercase. For example, from the String uni = “BiNus University”; when we use the method uni.toUpperCase(); and output it, the output will be “BINUS UNIVERSITY”, notice that all the characters are uppercase characters.

7. It is possible for different types of numeric values to be used together in computation as long however only one type will be returned with the value after the computation occurs.

8. i) a = 15
   ii) a = 48
   iii) a = 1
   iv) b = 7.5

9. i) 25 / 4 is 6
   ii) 25 / 4.0 is 6.25
   iii) 3 * 2 / 4 is 1
   Iv) 3.0 * 2 / 4 is 1.5

10.
```java
class test{
    public static void main(String[] args){
        int i = 0;
        int k = 100;
        int j = i + 1;

        System.out.println("j is " + j + " and k is " + k );
    }
}
```

11. i) char c = ‘A’ when converted with casting results in int i = 65.
    ii) float f = 10000.34F when converted with casting results in int i = 1000.
    iii) double d = 1000.34 when converted with casting results in int i = 1000.
    iv) int i = 97 when converted with casting results in a char c = ‘a’;

12. i) false
    ii) false
    iii) true
    Iv) true

13. i) Step 1: 2 * 2 - 3 > 2 && 4 - 2 > 5
2 * 2 is 4 (leftmost multiplication)

Step 2: 4 - 3 > 2 && 4 - 2 > 5
4 - 3 is 1 (leftmost subtraction)

Step 3: 1 > 2 && 4 - 2 > 5
4 - 2 is 2 (leftmost subtraction)

Step 4: 1 > 2 && 2 > 5
1 > 2 is false (leftmost greater than value comparison)

Step 5: false && 2 > 5
2 > 5 is false (leftmost greater than value comparison)

Step 6: false && false
false && false is false (last logical conjunction)

Step 7: false Hence, the expression is false

ii) Step 1: 2 * 2 - 3 > 2 || 4 - 2 > 5
2 * 2 is 4 (leftmost multiplication)

Step 2: 4 - 3 > 2 || 4 - 2 > 5
4 - 3 is 1 (leftmost subtraction)

Step 3: 1 > 2 || 4 - 2 > 5
4 - 2 is 2 (leftmost subtraction)

Step 4: 1 > 2 || 2 > 5
1 > 2 is false (leftmost greater than value comparison)

Step 5: false || 2 > 5
2 > 5 is false (leftmost greater than value comparison)

Step 6: false || false
false || false is false (last logical disjunction)

Step 7: false Hence, the expression is false