1. Write an application to determine the length of the string str=”Hello World”.(Hint Use String method)

Ans: **public** **class** Demo {

**public** **static** **void** main(String[] args) {

String str="Hello World";

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

}

}

1. Write an application to join two strings “Hello” and “How are you”. ”.(Hint Use String method)

Ans: **public** **class** Demo {

**public** **static** **void** main(String[] args) {

String str1=**new** String("Hello");

String str2=**new** String("How are you");

String s=String.*join*(" ", str1,str2);

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

}

}

1. Given a String ”Java String pool refers to collection of Strings which are stored in heap memory”, perform the following operations(Hint: all operation can be performed using string method.
2. Print string to console in lowercase

Ans  **public** **class** Demo {

**public** **static** **void** main(String[] args) {

String str1=" Java String pool is refers to colection of String

Which are stored in heap memory”;

String s1=str1.toLowerCase();

System.***out***.println(s1);

}

}

1. Print a string to console in uppercase

Ans: String s3=str1.toUpperCase();

System.***out***.println(s3);

1. Replace all ‘a’ character in string with ‘$’

Ans: String s1=str1.replace(“a”,”$);

1. Check if the original string contain the word “collection”.

Ans: System.***out***.println(str1.contains("collection"));

1. Check if the following string “java string pool refers to collection of

strings which are stored in heap memory” matches the string.

String str2="java string pool is refers to collection of strings

which are stored in heap memory”

System.***out***.println(str1.equals(str2));

1. If the string does not match check if there is another method which can be used to check if the string are equal

ans) System.***out***.println(str1==str2);

Assignments to StringBuffer Class

Note: StringBuffer is a peer class of String that provides much of the functionality of strings. String represents fixed-length, immutable character sequences while StringBuffer represents growable and writable character sequences. StringBuffer may have characters and substrings inserted in the middle or appended to the end.It will automatically grow to make room for such additions and often has more characters preallocated than are actually need,to allow room for growth.

1. Write an application to append the following strings “StringBuffer”,”is apeer class of String”, ”that provides much of”, ”the functionality of strings” using a StringBuffer.

Ans **public** **class** Demo {

**public** **static** **void** main(String[] args) {

StringBuffer sb=**new** StringBuffer();

String s1="StringBuffer";

sb.append(s1);

String s2=" is a peer class of String";

sb.append(s2);

String s3=" that provide much of";

sb.append(s3);

String s4=" the functionalities of string";

sb.append(s4);

System.***out***.println(sb.toString());

}

}

1. Insert the following string “insert text” into the string ”It is used to \_at the specified index position” at the location denoted by the sign \_

Ans) **public** **class** Demo {

**public** **static** **void** main(String[] args) {

StringBuffer sb=**new** StringBuffer("It is used to \_at the

Specified index position”)

String str="insert text";

sb.insert(14, str);

System.***out***.println(sb.toString());

}

}

1. Reverse the following string “This method returns the reversed object on which it was called” using StringBuffer Class

Ans **public** **static** **void** main(String[] args) {

StringBuffer sb=new StringBuffer(“This method returns object

On which it was called”)

sb.reverse();

System.***out***.println(sb.toString());

}

Assignment on StringBuider Class

Note:StringBuilder:J2SE 5 adds new string class to Java’s already powerful string handling capabilities This new class is called StringBuider. It is identical to StringBuffer except for one important difference:it is not synchronized,which means that it is not thread safe.The advantage of StringBuilder is faster performance.However, incase in which you are using multithreading,you must use StringBuffer rather than StringBuilder.

1) Write an application to append the following strings “J2SE adds new string”,”class to Java’s already powerful string”, ”handling capabilities”, ”this new class is StringBuilder” using a StringBuilder.

Ans) **public** **static** **void** main(String[] args) {

StringBuilder sb=**new** StringBuilder();

String s1="J2SE adds new string";

sb.append(s1);

String s2=" class to Java’s already powerful string";

sb.append(s2);

String s3=" handling capabilities";

sb.append(s3);

String s4=" this new class is String Builder";

sb.append(s4);

System.***out***.println(sb.toString());

}

1. Insert the following string “insert text” into the string ”It is used to \_at the specified index position” at the location denoted by the sign \_

Ans) **public** **class** Demo {

**public** **static** **void** main(String[] args) {

StringBuilder sb=**new** StringBuilder("It is used to \_at the

Specified index position”)

String str="insert text";

sb.insert(14, str);

System.***out***.println(sb.toString());

}

}

1. Reverse the following string “This method returns the reversed object on which it was called” using StringBuilder Class

Ans **public** **static** **void** main(String[] args) {

StringBuffer sb=new StringBuffer(“This method returns object

On which it was called”)

sb.reverse();

System.***out***.println(sb.toString());

}