

**FACULTY:** APPLIED SCIENCES

**DEPARTMENT:** COMPUTER SCIENCE

**COURSE:** OBJECT ORIENTED PROGRAMMING IN JAVA

**COURSE CODE:** SCS 2108

**YEAR:** [PART 2] SEMESTER 1

**NAME** THEMBEKILE B NDEBELE

**STUDENT NUMBER** N02018976L

**TEST** 1

QUESTION 1

1.Write a Java program that implements Socket Programming [15]  
  
SOLUTION  
CLIENT-SIDE CODE

package ClientSocket**;**

**import** java**.**io**.**BufferedReader**;**

**import** java**.**io**.**BufferedWriter**;**

**import** java**.**io**.**IOException**;**

**import** java**.**io**.**InputStreamReader**;**

**import** java**.**io**.**OutputStreamWriter**;**

**import** java**.**io**.**PrintStream**;**

**import** java**.**net**.**Socket**;**

**import** java**.**util**.**Scanner**;**

public class client **{**

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

Socket socket**=** **null;**

InputStreamReader inputStreamReader **=** **null;**

OutputStreamWriter outputStreamWriter **=** **null;**

BufferedReader bufferedReader **=** **null;**

BufferedWriter bufferedWriter **=** **null;**

**try**

**{**

socket **=** **new** Socket**(**"localhost"**,**1234**);**

inputStreamReader **=** **new** InputStreamReader**(**socket**.**getInputStream**());**

outputStreamWriter **=** **new** OutputStreamWriter**(**socket**.**getOutputStream**());**

bufferedReader **=** **new** BufferedReader**(**inputStreamReader**);**

bufferedWriter **=** **new** BufferedWriter**(**outputStreamWriter**);**

Scanner scanner **=** **new** Scanner**(**System**.**in**);**

**while(true){**

String msgToSend **=** scanner**.**nextLine**();**

bufferedWriter**.**write**(**msgToSend**);**

bufferedWriter**.**newLine**();**

bufferedWriter**.**flush**();**

System**.**out**.**println**(**"Server: " **+** bufferedReader**.**readLine**());**

**if** **(**msgToSend**.**equalsIgnoreCase**(**"BYE"**))**

**break;**

**}}**

**catch(**IOException e**){**e**.**printStackTrace**();}**

**finally** **{**

**try{**

**if** **(**socket **!=** **null)** socket**.**close**();**

**if** **(**inputStreamReader **!=** **null)** inputStreamReader**.**close**();**

**if** **(**outputStreamWriter **!=** **null)** outputStreamWriter**.**close**();**

**if** **(**bufferedReader **!=** **null)** bufferedReader**.**close**();**

**if** **(**bufferedWriter **!=** **null)** bufferedWriter**.**close**();**

**}**

**catch(**IOException e**)** **{**e**.**printStackTrace**();}**

**}**

**}}**

SERVER-SIDE CODE

package ServerSocket**;**

**import** java**.**io**.**BufferedReader**;**

**import** java**.**io**.**BufferedWriter**;**

**import** java**.**io**.**IOException**;**

**import** java**.**io**.**InputStreamReader**;**

**import** java**.**io**.**OutputStreamWriter**;**

**import** java**.**net**.**ServerSocket**;**

**import** java**.**net**.**Socket**;**

public class server **{**

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

Socket socket **=** **null;**

InputStreamReader inputStreamReader **=** **null;**

OutputStreamWriter outputStreamWriter **=** **null;**

BufferedReader bufferedReader **=** **null;**

BufferedWriter bufferedWriter **=** **null;**

ServerSocket serverSocket **=** **null;**

serverSocket **=** **new** ServerSocket**(**1234**);**

**while(true)** **{**

**try{**

socket **=** serverSocket**.**accept**();**

inputStreamReader **=** **new** InputStreamReader**(**socket**.**getInputStream**());**

outputStreamWriter **=** **new** OutputStreamWriter**(**socket**.**getOutputStream**());**

bufferedReader **=** **new** BufferedReader**(**inputStreamReader**);**

bufferedWriter **=** **new** BufferedWriter**(**outputStreamWriter**);**

**while(true)** **{**

String msgFromClient **=** bufferedReader**.**readLine**();**

System**.**out**.**println**(**"Cient: " **+**msgFromClient**);**

bufferedWriter**.**write**(**"MSG Received."**);**

bufferedWriter**.**newLine**();**

bufferedWriter**.**flush**();**

**if** **(**msgFromClient**.**equalsIgnoreCase**(**"BYE"**))**

**break;**

**}**

socket**.**close**();**

inputStreamReader**.**close**();**

outputStreamWriter**.**close**();**

bufferedReader**.**close**();**

bufferedWriter**.**close**();**

**}**

**catch(**IOException e**){**e**.**printStackTrace**();** **}**

**}**

**}**

**}**

QUESTION 2

2.Explain how a predicate can be used in java message service [10]  
  
ANSWER

Java Message Service) is an API that provides the facility to create, send and read messages.  
Predicate is a functional interface defined in java.util.function package which can be used in all the contexts where an object needs to be evaluated for a given test condition and a boolean value needs to be returned based on whether the condition was successfully met or not.

Predicates can be used to filter lists in messages.  
  
Example:  
  
**public** **static** List**<**String**>** taskTitles**(**List**<**Task**>** tasks**,** Predicate**<**Task**>** filterTasks**)** **{**

List**<**String**>** readingTitles **=** **new** ArrayList**<>();**

**for** **(**Task task **:** tasks**)** **{**

**if** **(**filterTasks**.**test**(**task**))** **{**

readingTitles**.**add**(**task**.**getTitle**());**

**}**

**}**

**return** readingTitles**;**

**}**

This code snippet uses a predicate to filter out reading tasks.  
  
Predicates can also use multiple conditions to filter integers.In the case of student marks, two conditions can be passed to view marks above a certain value:  
  
Predicate**<**Student**>** p1 **=** s **->** s**.**name**().**startsWith**(**"I"**)** **&&**

s**.**mark**()** **>** 49**;**