### EEEM023 / EEM.nsm Lab-Based Assignment

The assignment consists of practical computer laboratory work, with the aim of helping you to gain familiarity with SNMP in a manager role. Software is to be developed and submitted, together with a brief report covering the areas addressed in the practical work. Every student will also give in the laboratory a demonstration of the software that he or she has built.

이 과제는 매니저 역할의 SNMP에 익숙해지는 것을 목적으로 한 실용적인 컴퓨터 실험실 작업으로 구성됩니다.

All the programming required should be done in **Java**.

The following two small systems should be developed:

* A manager program that retrieves via the Advent SNMP low level API all the TCP connections in a remote machine. The TCP connection objects should be as defined in the SNMP MIB-II.
* A manager program that is given as parameters a counter attribute (i.e. a SNMP “object”) that models real traffic to be monitored, e.g. *ifInOctets* / *ifOutOctets* of the SNMP MIB-II Interface group, *ipInDelivers* / *ipOutRequests* of the IP group, *tcpInSegs* / *tcpOutSegs* of the TCP group, etc., a polling period in seconds, the number of periods *N* to be used for averaging, and the network node in which to access that object. The program should poll the object periodically and should calculate the throughput, in bits/sec or packets/sec, using a uniformly weighted moving average algorithm (the latter is explained in the course notes).
* 모니터 대상 실제 트래픽을 모델화하는 카운터 속성(즉 SNMP "개체")이 파라미터로 주어지는 매니저 프로그램만들기. 예를 들어 SNMP MIB-II 인터페이스 그룹의 ifInOctets / ifOutOctets, IP 그룹의 ipInDelivers / ipOutRequests, tcpInSe tcpOuts / tcpOutgse 등입니다. 폴링 기간(초단위), 평균화에 사용되는 기간 N 및 해당 오브젝트에 액세스하는 네트워크노드 , 프로그램은 오브젝트를 정기적으로 폴링하고 균등 가중치 이동 평균 알고리즘을 사용하여 스루풋을 비트/초 또는 패킷/초 단위로 계산해야 합니다(후자는 코스 노트에 설명되어 있습니다).

Please avoid printing documentation, and instead read it online. Support laboratory sessions are run between Weeks 4 and 9 to help you with the work. Please refer to your timetable. You should start working before then so that you have meaningful questions and interaction during the sessions. Finally, lab guidelines with useful information and resources are provided in Appendixes I.

### Deliverables

There are two deliverables:

1. You **must** show your software to one of the demonstrators at one of the lab sessions in Week 9. In case of remote labs please send your software through SurreyLearn. If necessary you will be asked to have an online session to explain your code together with other Q&As.
2. You **must** submit via SurreyLearn a brief report on your work by the Week 12 deadline given on the front of this sheet, jointly with other assignment assessment components.

### Report structure

The report should be brief, of the order of 4-5 pages, and consist of the following:

* An outline of your program design, in the form of pseudo-code or a flowchart; together with brief notes on any key features of the software; and on any issues encountered during your work together with their resolution;
* Relevant screenshots of your program’s output;
* Validation of the UWMA results you have obtained;
* Concluding comments.

Do not include your software source code in the report

### Assessment criteria

Software:

* Functions implemented correctly
* Quality of output display (correct output of TCP connection table; correct display of UWMA parameters and calculation of UWMA)
* Code clarity and appropriate commenting
* Your ability to discuss your design and implementation of your code
* • 올바르게 구현된 기능
* • 출력 표시 품질(TCP 접속 테이블의 올바른 출력, UWMA 파라미터의 올바른 표시 및 UWMA의 계산)
* • 코드의 명확성과 적절한 코멘트
* • 코드 설계 및 구현에 대해 논의할 수 있는 능력

Report:

* Quality of SNMP discussion
* Quality of software description
* Results / screenshots
* Quality of English

### Appendix – For Java developer

Below are some guidelines that would be useful in the coursework. Note that the lab demonstrators will show you step by step of the following processes in the first lab session.

Eclipse:

* Eclipse is an IDE (Integrated Developing Environment) that you will use to develop your Java programs. To run Eclipse, go to "Applications" 🡪 "Programming" 🡪 "Eclipse". Use the default workspace. Basically, the steps are:

Create new Java project 🡪 Create new .java file under "src" directory 🡪 Start programming 🡪 Compile 🡪 Run (if successful) or debug (if unsuccessful).

* Another thing you need to know is how to pass command-line arguments in Eclipse.

Firstly, open the .java file you want to run. Then, select "Run" in the menu bar 🡪 "Run configurations". Now, check whether the name of your class appears under "Java Application" on the left. If no, then you need to double-click on "Java Application" to see it appear. Normally you only need to do this once. After this is done, select "Arguments" tab on the right. Enter the command-line arguments you want to input under "Program arguments", each separated by a space.

AdventNet SNMP Package:

* With Java itself, we cannot perform SNMP network operations. Therefore, we need an external package as a complement to Java to do this coursework, which is "AdventNet SNMP" package. This package contains many Java classes which are capable of sending SNMP requests, retrieving SNMP entries, displaying information etc.
* We need to link this package to out Java project via Eclipse. The steps to link the package are:

Download the AdventNetSnmp package [here](http://info.ee.surrey.ac.uk/CCSR/Internal/Networks/NSM-coursework/AdventNetSnmp.jar) 🡪 Right click on your Java project folder 🡪 "Build Path" 🡪 "Select external archives" 🡪 Select the AdventNetSnmp.jar file, then "OK".

Example Java Programmes:

* Two example Java programmes are provided to illustrate how one can use AdventNetSnmp package to perform SNMP operations:

1) Send a GET request for a specified OID value: [SnmpGet.java](http://info.ee.surrey.ac.uk/CCSR/Internal/Networks/NSM-coursework/SourceCode/SnmpGet.java)

2) Send a GETNEXT request for a specified OID value: [SnmpGetNext.java](http://info.ee.surrey.ac.uk/CCSR/Internal/Networks/NSM-coursework/SourceCode/SnmpGetNext.java)

* More explanations on these two programs can be found by going [here](http://info.ee.surrey.ac.uk/CCSR/Internal/Networks/NSM-coursework/help/index.html), and select "Tutorials -> Low Level API Tutorials -> SNMP Get and Get Next". Also, we will explain SnmpGetNext.java step by step in the first lab session.

Hosts used in this lab:

If we want to get an SNMP response, we need to send the requests to a host that runs an SNMP agent on it. The current host which has an SNMP agent installed is:

* Host: feps-teach01
* Community: teachinglabs

OIDs used in this lab:

First of all, make sure that you understand what an OID is, and why it looks like this.

* For the first part, the following five OIDs will be used:

.1.3.6.1.2.1.6.13.1.1

.1.3.6.1.2.1.6.13.1.2

.1.3.6.1.2.1.6.13.1.3

.1.3.6.1.2.1.6.13.1.4

.1.3.6.1.2.1.6.13.1.5

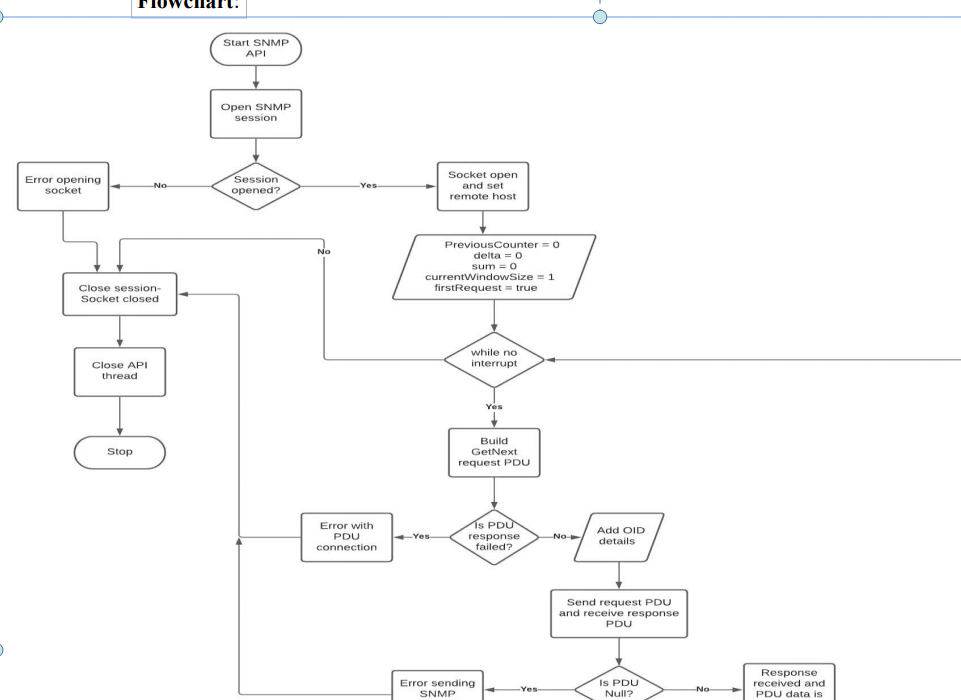
* For the second part, the following OID pairs should be used:

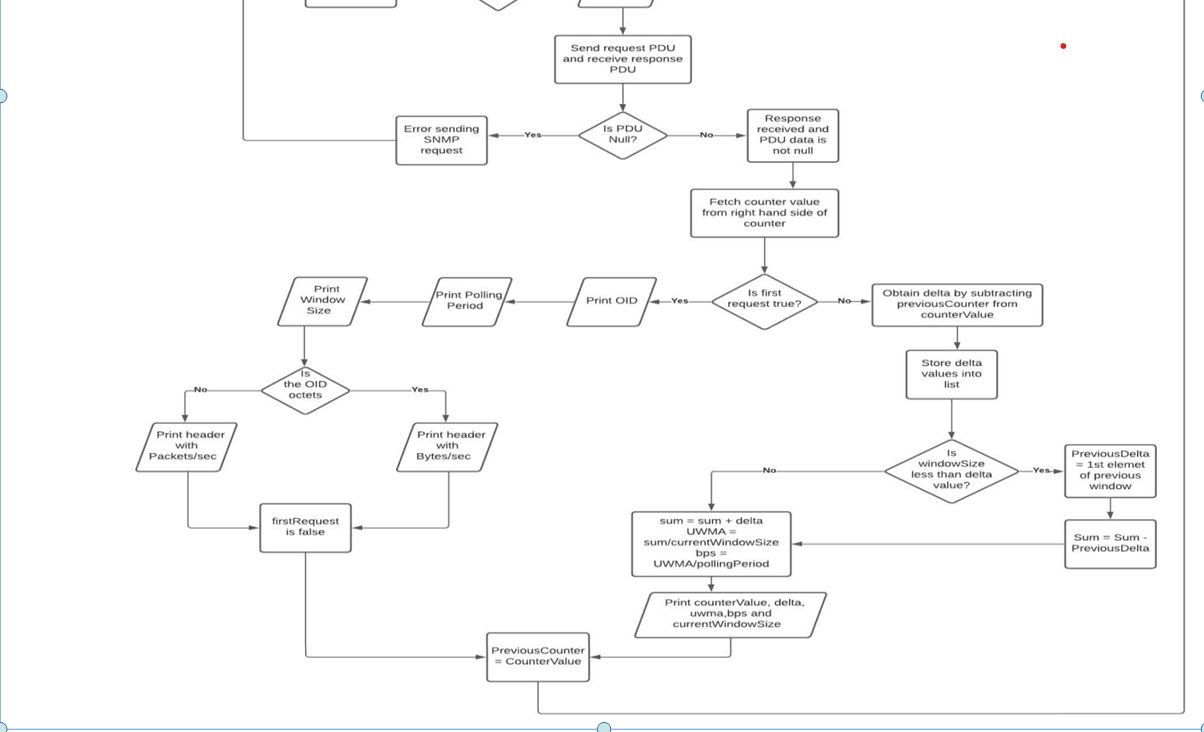
.1.3.6.1.2.1.6.10 and .1.3.6.1.2.1.6.11 (tcpInSegs and tcpOutSegs)

.1.3.6.1.2.1.2.2.1.10 and .1.3.6.1.2.1.2.2.1.16 (ifInOctets and ifOutOctets)

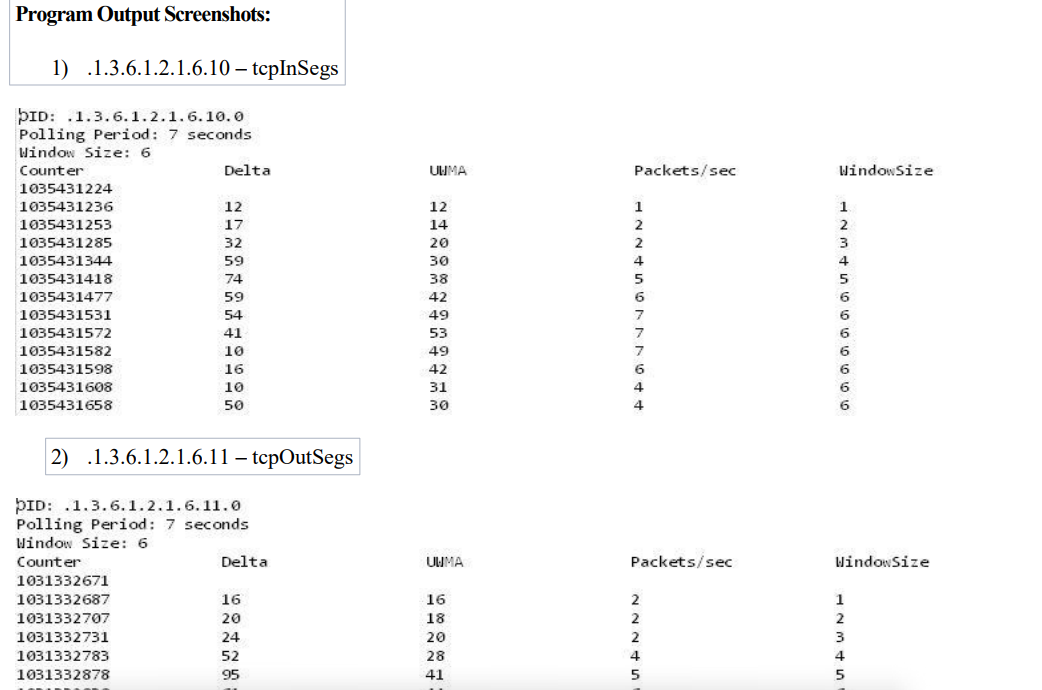
.1.3.6.1.2.1.4.9 and .1.3.6.1.2.1.4.10 (ipInDelivers and ipOutRequests)

<플로우차트>





<원하는출력>



위에 세줄까지는 출력이되는데 밑에 테이블까지 출력되게 코드수정해야합니당 ㅠㅠㅠ