Práctica 1 - Parte 3

Objetivos:

- Implementar operaciones SNMP get y set para adquirir información de variables de la MIB
- Almacenar la información adquirida de manera periódica.
- Mostrar la información almacenada dentro de un intervalo de tiempo.

Tarea 1 Usar pySNMP para adquirir la información de administración

Instalación por línea de comando:
La mejor manera de obtener PysSNMP y sus dependencias es usando PIP v2

```
$ pip install pysnmp
```

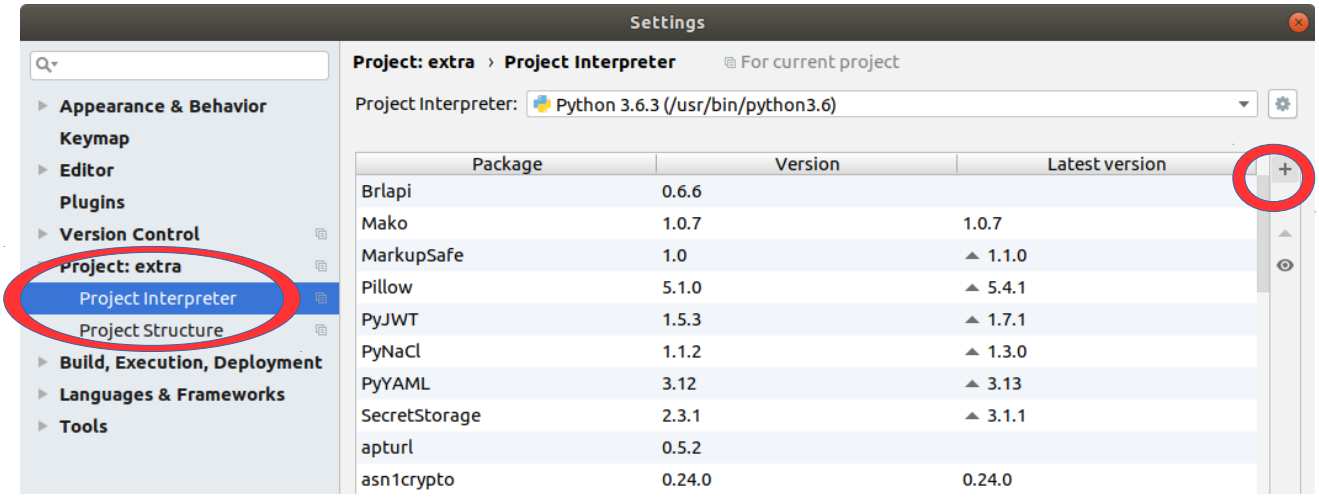
o PIP v3

```
$ pip3 install pysnmp
```

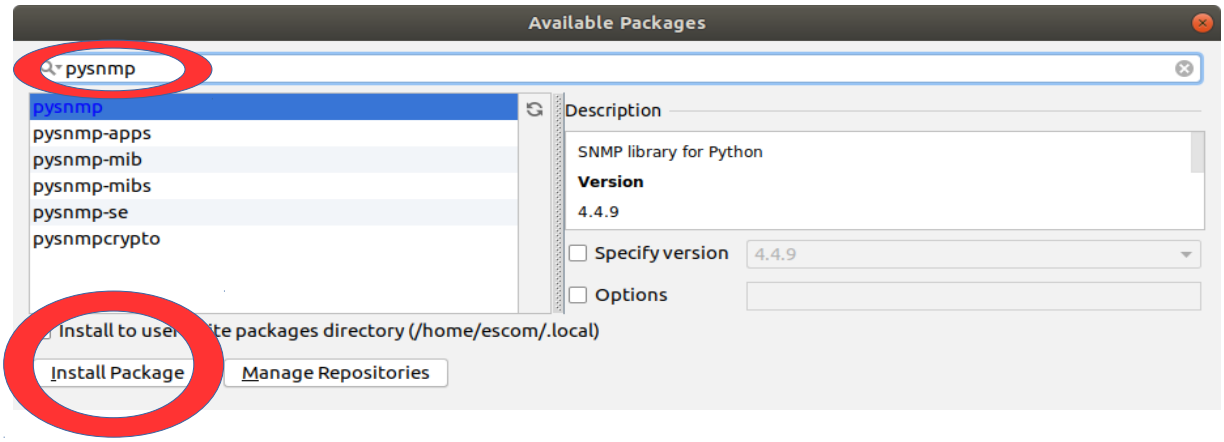
Instalación usando IDE Pycharm 2018.3.4 (Community Edition)

$ sudo snap install pycharm-community –classic

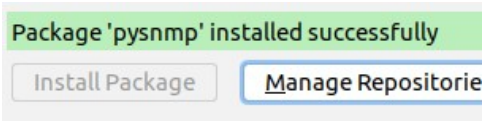
En el entorno gráfico, menú File, Settings.
1) Seleccionar Las opciones del proyecto, sub menú Proyect Interpeter
2) Clic en instalar (+)



Buscar PySNMP e instalar el paquete.



Verificar la instalación



Ejercicio 1 (sello 1)
Descargar el código v1-get.py del repositorio de github https://github.com/dratani/Practica1. Analizar el comando getCmd, revisar la documentación y responder el siguiente cuestionario
http://snmplabs.com/pysnmp/docs/hlapi/asyncore/sync/manager/cmdgen/getcmd.html
1) ¿Cuál es el objetivo del comando getCmd?
2) ¿Qué representa el parámetro snmpEngine y escribe los valores que acepta?
3) ¿Para qué sirve el parámetro transportTarget?
4) ¿Cuál es la función del parámetro contextData?
5) Describe el parámetro varBinds
Ahora, analiza los datos de salida del comando getCmd y responde.
6) ¿Para qué sirve el campo errorIndication?, escribe su tipo de dato.
7) ¿Qué información proporciona errorStatus?, escribe su tipo de dato.
8) ¿Cuál es la función de errorIndex?, escribe su tipo de dato.
9) ¿Cuál es el objetivo de varBinds?, escribe su tipo de dato.

Ejercicio 2 (sello 2)
Modificar el script v1-get.py, colocar la información de alguno de los agentes que configuró en sesiones anteriores. Ejecutar el script y mostrar el resultado.

Ejercicio 3 (sello 3)
Descargar el archivo v2c-get.py del repositorio de github https://github.com/dratani/Practica1, explicar las diferencias entre la implementación de la versión uno (v1-get.py) y la versión 2 (v2c-get.py)

Modificar el script v2c-get.py, colocar la información de alguno de los agentes que configuró en sesiones anteriores. Ejecutar el script y mostrar el resultado.

Tarea 2 usar rrdtool para almacenar la información de administración

Requerimientos
Antes de instalar, asegurar que las dependencias están instaladas, de otra manera el módulo fallará.

- Python 2.6+, 3.3+
- Python header y archivos de librerías
- librrd header y archivos de librerías.

En debian/ubuntu, para generar los enlaces en Python 2.X usa:

```
$ sudo apt-get install librrd-dev libpython-dev
```

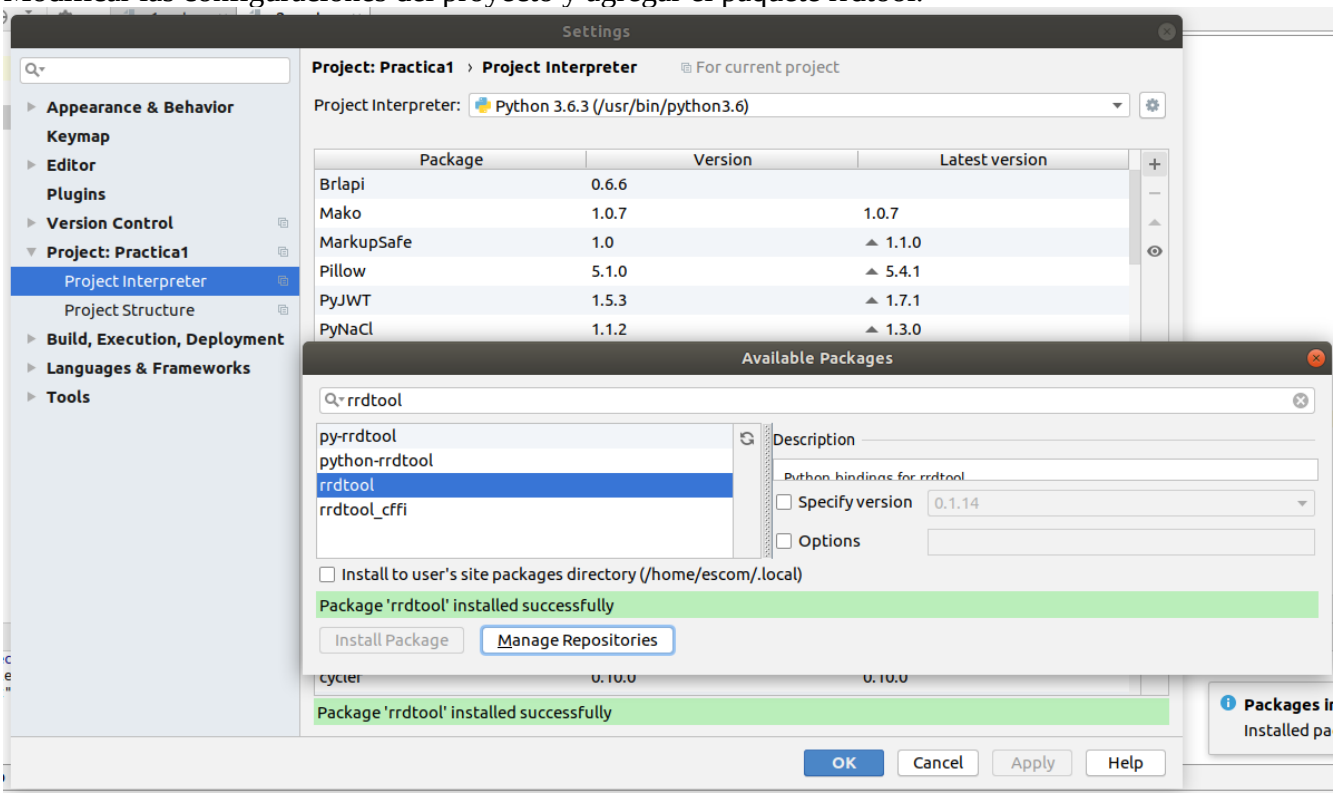
Para Python 3.X usa

```
$ sudo apt-get install librrd-dev libpython3-dev
```

Instalación en línea de comandos
La forma más fácil es usar pip (usar la versión acorde a python):

```
$ pip install rrdtool
```

Instalación usando el IDE Pyharm
Modificar las configuraciones del proyecto y agregar el paquete rrdtool.



Para otros sistemas operativos revisar la documentación https://pythonhosted.org/rrdtool/index.html

Ejercicio 4. (sello 4)
Descargar y analizar el script sample.py del repositorio de github https://github.com/dratani/Practica1. Consultar la documentación oficial https://oss.oetiker.ch/rrdtool/ y describir las funciones create, update, dump, fetch y graph. Ejecutar el script y mostrar el resultado.

Tarea 3 - Módulo de adquisición de información usando pySNMP y rrdtool

Ahora, hay que juntar los dos módulos para adquirir y almacenar información de administración en tiempo real.
Descargar los archivos createRRD.py updateRRD.py y graphRRD.py del repositorio de github https://github.com/dratani/Practica1