**Complejidad computacional del algoritmo:**

Analizando el código realizado para la resolución del problema:

**public void seleccionarAplicaciones**() {

**int cantAppSumadas**;

**int valorSumaApps**;

**for** (**int i** = 0;i < **this**.cantidadDeAplicaciones; i++) {

cantAppSumadas = 0;

valorSumaApps = 0;

**int j** = i;

**while** (valorSumaApps < **this**.tamañoNuevaAplicacion && j < **this**.cantidadDeAplicaciones) {

cantAppSumadas++;

valorSumaApps += **this**.tamañosAplicaciones.get(j);

j++;

}

**if** (valorSumaApps < **this**.tamañoNuevaAplicacion) {

**break**;

}

**AplicacionesADesinstalar appsADesinstalar** = **new** AplicacionesADesinstalar(cantAppSumadas, valorSumaApps, i + 1);

**this**.aplicaciones.add(appsADesinstalar);

}

**if** (!**this**.aplicaciones.isEmpty())

**Collections**.*sort*(**this**.aplicaciones);

}

Se llega a la conclusión de que la complejidad computacional del algoritmo es aproximadamente O(n2), ya que se itera sobre dos bucles anidados. Aclaramos que la complejidad no llega a exactamente n2 ya que el bucle interno no itera sobre todos los elementos del arreglo, pero no es inferior a n.log(n).

Se puede ver el gráfico de rendimiento para ver la línea de tendencia del algoritmo, junto con su ecuación de la curva de regresión polinómica de grado 2, la cual se acerca bastante al rendimiento obtenido en el gráfico.