

**Profesor**Diego Seco

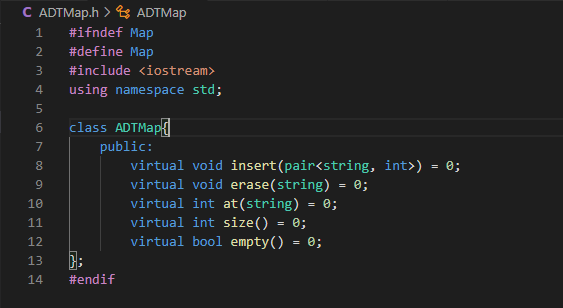
Universidad de Concepción  
Facultad de Ingeniería

Departamento de Ing. Informática y Cs de la Computación

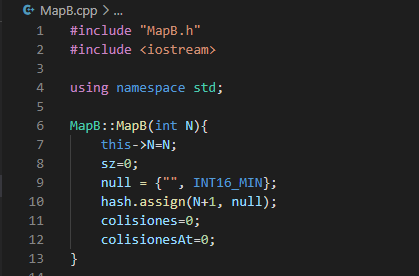
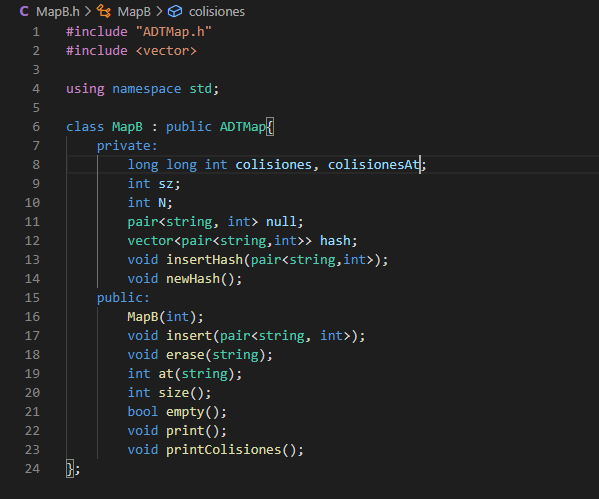
**Laboratorio 6:**Estructuras de Datos (503220)

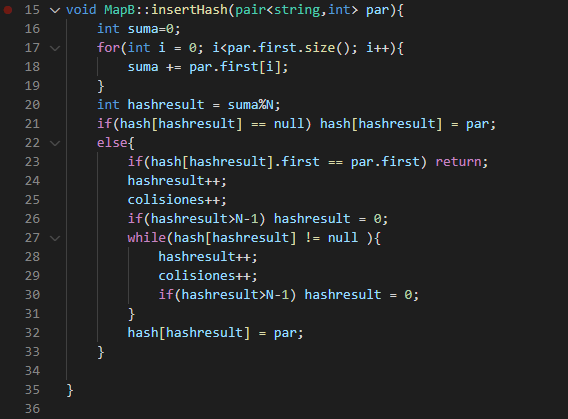
**Estudiante:** Nicolás Araya (2018448613)

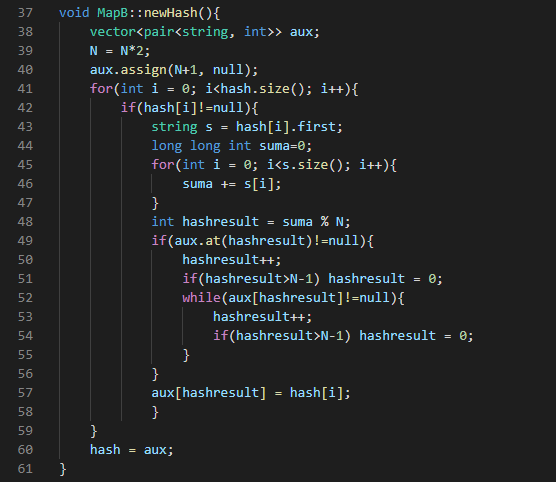
1. **ADTMap:**

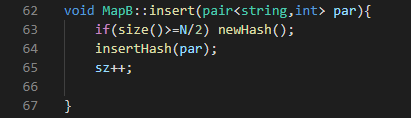


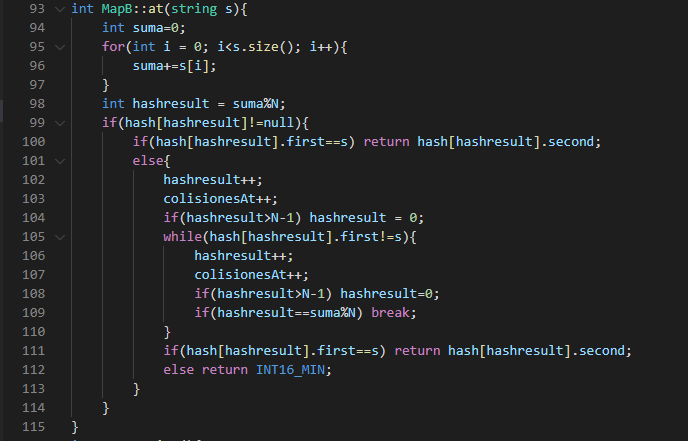
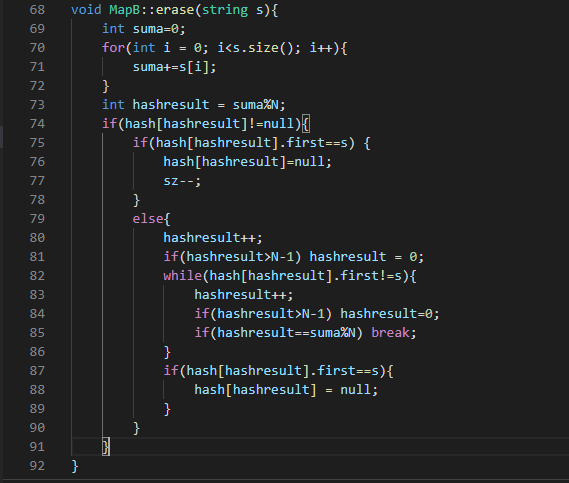
1. **MapB:**

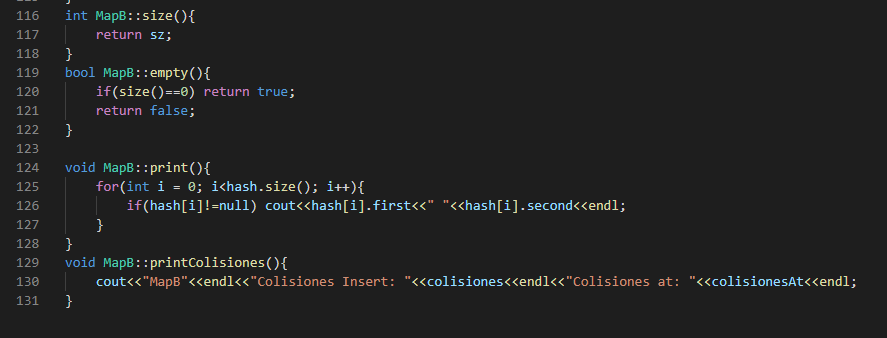






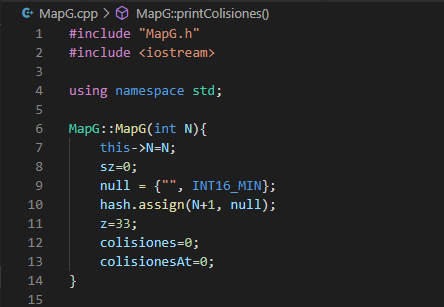




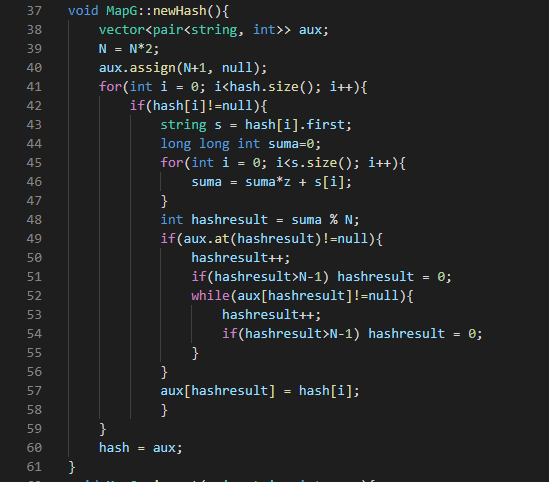


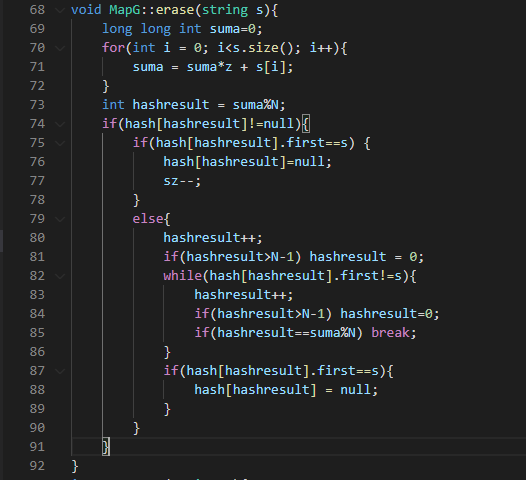
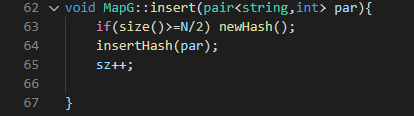
**MapG:**

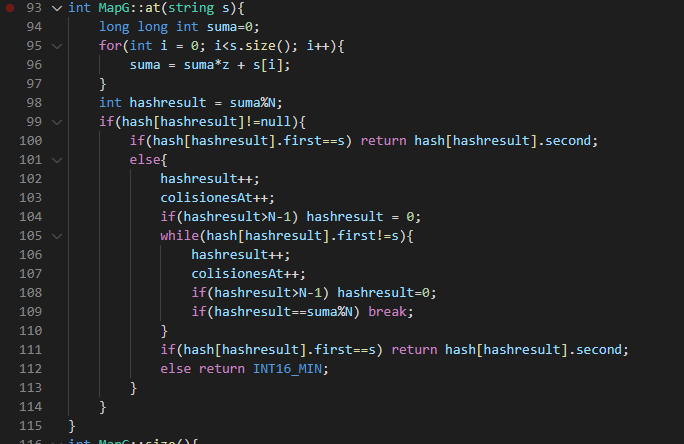


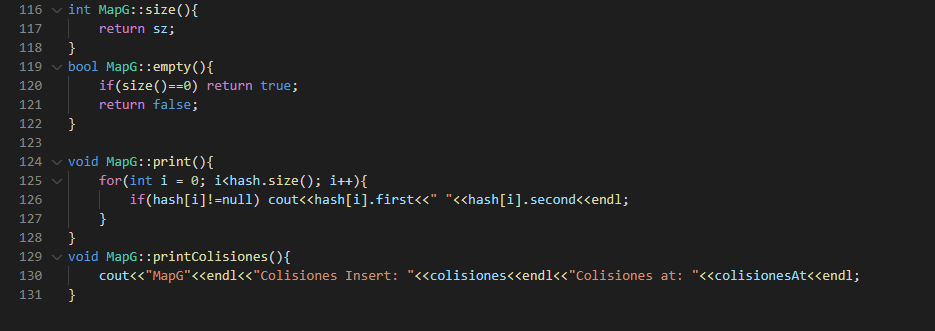




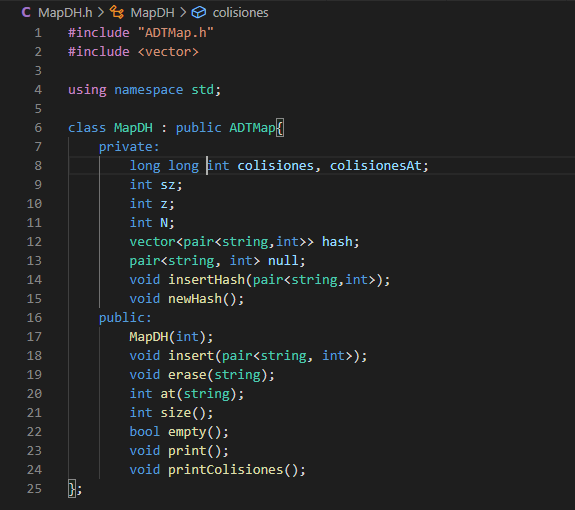


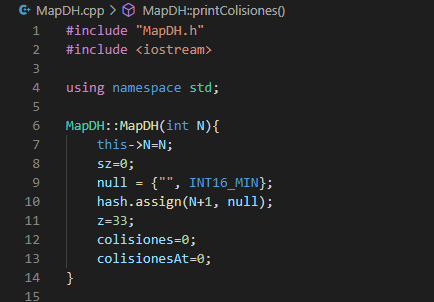


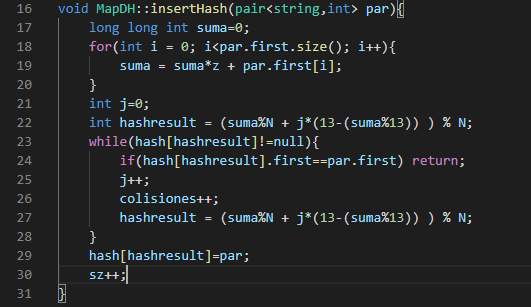


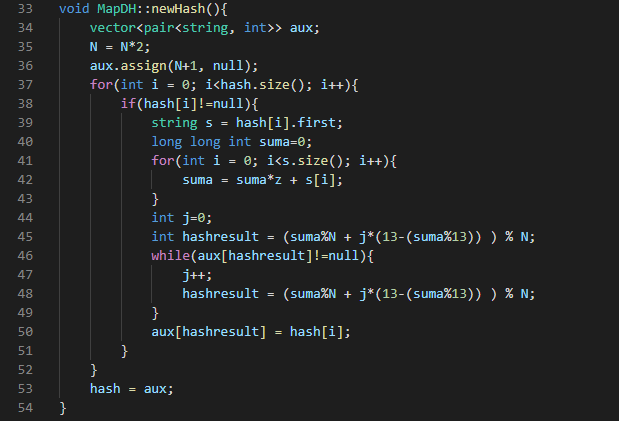


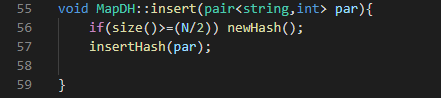
**MapDH:**

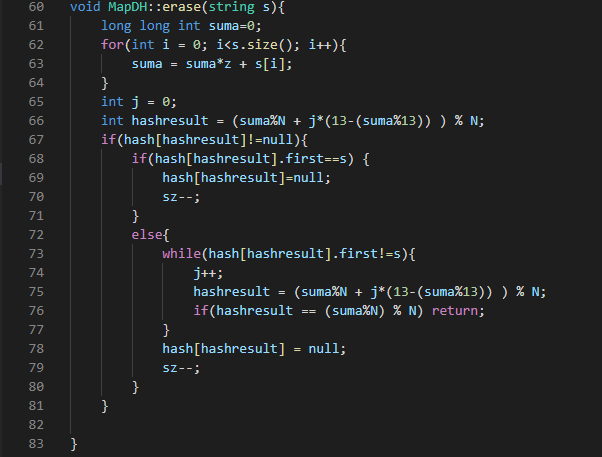


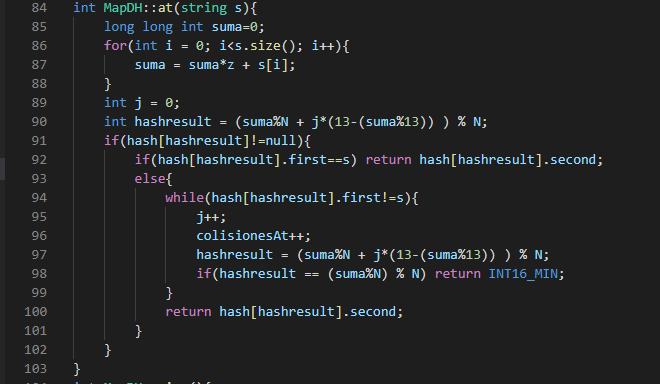


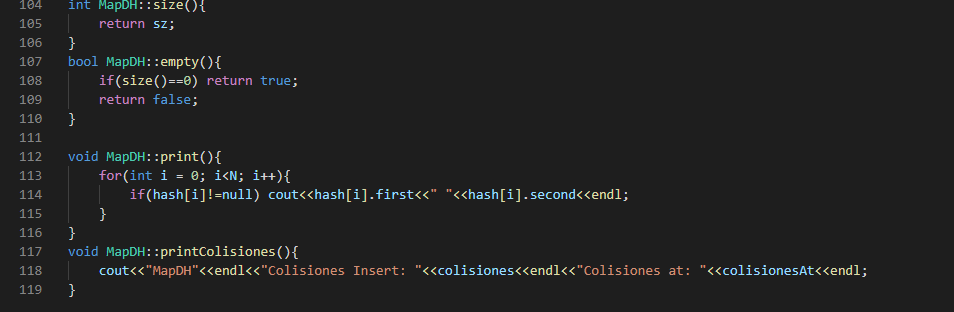












**3)**

**Método Insert:**

|  |  |  |  |
| --- | --- | --- | --- |
| N | Número de colisiones en Insert | | |
|  | **MapB** | **MapG** | **MapDH** |
| 2000 | **1912923** | **1312** | **1160** |
| 4000 | **7810661** | **2855** | **2289** |
| 6000 | **17725295** | **4529** | **3415** |
| 8000 | **31613341** | **5891** | **4756** |
| 10000 | **49500933** | **7417** | **5838** |

**Método at:**

|  |  |  |  |
| --- | --- | --- | --- |
| N | Número de colisiones en at | | |
|  | **MapB** | **MapG** | **MapDH** |
| 2000 | **1912923** | **900** | **772** |
| 4000 | **7810661** | **1913** | **1530** |
| 6000 | **17722446** | **3016** | **2379** |
| 8000 | **31603103** | **3961** | **3260** |
| 10000 | **49466358** | **5047** | **4039** |

Análisis Teórico:

Los métodos insert y at, en pero caso son del orden O(n) ya que en peor caso podrían generarse n-1 colisiones. En todo caso se trata de un caso muy puntual y especifico. Al realizar rehashing, el tiempo es O(n+m), con m igual al nuevo espacio añadido. Considerando que cada vez se duplica el espacio este es O(2n).

Análisis experimental:

De los gráficos y datos tabulados se puede concluir que el MapB, es una forma muy mala de implementar un Map, principalmente por la función Hash implementada, la cual se basaba en sumar cada componente del string, lo que produce una gran cantidad de colisiones. También se puede concluir la importancia de utilizar una buena función de Hash comparando el MapB y MapG ya que, pese a que ambos utilizan linear probing, la buena función de Hash utilizada en MapG (Acumulación polinomial) hizo disminuir la cantidad de colisiones en gran medida. Y respecto a MapDH, el cual pasa del linear probing al doble hashing, utilizando la misma función de Hash que en MapG. Las colisiones si bien son menores, tampoco hay una diferencia tan marcada. En conclusión, el factor más importante para la implementación de Map es utilizar una buena función de Hash, como lo es la acumulación polinomial.