**Algorithm analysis, Selection Sort**

|  |  |  |
| --- | --- | --- |
| **Instruction** | **Cost** | **# Repeat** |
| ArrayList<Game> list = c.getWishList(); |  |  |
| for (int i = 0; i < list.size(); i++) |  |  |
| Game minor = list.get(i); |  |  |
| int cual = i; |  |  |
| for (int j = i + 1; j < list.size(); j++) |  |  |
| Shelve s1 = searchShelve(list.get(i).getShelveName()); |  |  |
| Shelve s2 = searchShelve(list.get(j).getShelveName()); |  |  |
| if (s1.getNameShelve().compareTo(s2.getNameShelve()) == 0) |  |  |
| if (s2.getGameShelve().getIndexInTable(list.get(j).getCode()) < s1.getGameShelve()  .getIndexInTable(list.get(i).getCode())) |  |  |
| minor = list.get(j); |  |  |
| cual = j; |  |  |
| else if (s1.getNameShelve().compareTo(s2.getNameShelve()) > 0) |  |  |
| Game temp1 = list.get(i); |  |  |
| list.set(i, list.get(j)); |  |  |
| list.set(j, temp1); |  |  |
| Game temp2 = list.get(i); |  |  |
| list.set(i, minor); |  |  |
| list.set(cual, temp2); |  |  |
| c.setWishList(list); |  |  |

**Note:** We assume that

**Analysis of time complexity**

Assuming that

We have that

**Spatial complexity analysis**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Variable** | **Size of 1 atomic value** | **Amount values** |
| Input | List | 32 bits | 1 |
| Auxiliar | Minor | 32 bits | 1 |
| Auxiliar | Cual | 32 bits | 1 |
| Auxiliar | S1 | 32 bits | 1 |
| Auxiliar | S2 | 32 bits | 1 |
| Auxiliar | Temp1 | 32 bits | 1 |
| Auxiliar | Temp2 | 32 bits | 1 |

* Total Spatial Complexity
* Auxiliary Spatial Complexity
* Auxiliary Spatial Complexity Output