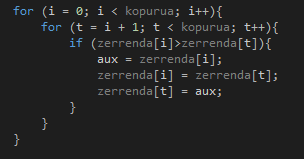
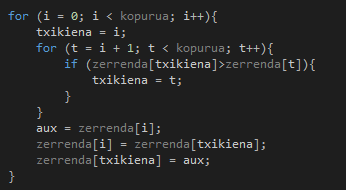
|  |
| --- |
| **Ordenazio algoritmoak** |
| Ordenazio algoritmo desberdinak eta konparaketa |
|  |
| Dokumentu honetan 8 ordenazio algoritmo aztertu eta haien arteko konparaketa egiten da. Hori eta gero denen artean dauden desberdintasunak argitu eta haietako onena erakusten da |
|  |
| **Aitor Urrutia Zubikarai** |
| **11/04/2014** |
|  |

1. Ordenazio metodoen analisia
   1. Hautaketa zuzena (selección directa)
      * 1. Azalpena

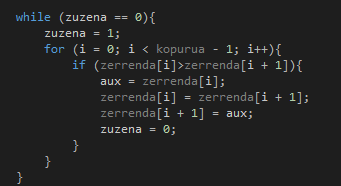
* Listako elementurik txikiena bilatzen du
* Listako lehenengo elementuarekin elkartrukatzen da
* Listako bigarren elementurik txikiena bilatzen da
* Listako bigarren elementuarekin trukatzen da
* Lista osoa ordenatu arte segitzen da
  + - 1. Algoritmoaren analisia
* Memoria eskakizunak: Burbuilaren metodoak bezala, aldagai auxiliar baten beharra dago.
* Exekuzio denbora: Kanpoko buklea n aldiz Egin behar da (n listako elementu kopurua izanik). Bukle horren barruan, momenturarte ordenatuta ez dauden elementu guztiak pasatu behar dira.
  + - 1. Kodigoa



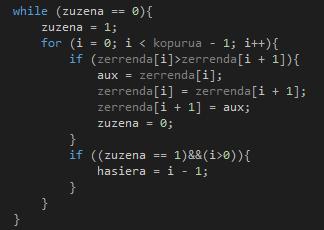
* + - 1. Abantailak
* Inplementazio erraza
* Ez du memoria gehiagoren beharrik
* Errendimendu konstantea (kasurik txarrenetik kasurik onenerainoko denbora diferentzia oso txikia da)
  + - 1. Desabantailak
* Geldoa
* Konparazio kantitate handia egiten du
  + - 1. Hobekuntzak
* Listako zenbakiak denekin konparatu eta aldaketa egiten da
* Horrela aldaketa kopurua handiegia da
* Hori sahiezteko konparaketa guztien amaieran egiten da aldaketa
* Konparaketa momenturarte dagoen zenbaki txikienarekin egiten da
* Hasieran, zenbaki txikiena bezala lehenengo zenbakia artzen da
* Kodigoa:



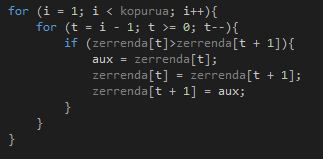
* 1. Burbuilaren Metodoa (Burbuja – bubleSort)
     + 1. Azalpena
* Lehenengo bi zenbakiak konparatzen dira, ordenatuta ez badaude elkartrukatu egiten dira
* Bigarren eta hirugarren zenbakiak konparatzen dira, ordenatuta ez badaude elkartrukatu egiten dira
* Horrela zerrenda guztia pasatzen da
* Zerrenda behin eta berriz pasatuko da aldaketarik egon ez arte
  + - 1. Algoritmoaren analisia
* Memoria eskakizunak: Aldagai auxiliar baten beharra dago.
* Exekuzio denbora: Ordenatu beharreko zerrenda oso desordenatuta baldin ez badago, barruko buklea gutxitan egiten da eta beraz irauten duen denbora ez da ain handia. Ordenatu beharreko zerrenda oso desordenatuta baldin badago, askotan egin behar izaten da barruko buklea eta exekuzio denbora asko luzatzen da.
  + - 1. Kodigoa



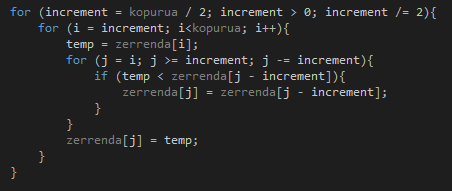
* + - 1. Abantailak
* Nahiko simplea da
* Kodigo motza da ordenazio algoritmo bat izateko
* Efikazia handia du
  + - 1. Desabantailak
* Denbora asko irauten du ordenaketa prozesuan
  + - 1. Hobekuntzak
* Algoritmo honen hobekuntza nahiko simplea da
* Bukle bakoitzaren hasieran hasi beharrean, aldaketak Egin diren lehenengo zatitik aurrera hasten da konparatzen



* 1. Txertaketa zuzena (Inserción directa)
     + 1. Azalpena
* Algoritmo mota hau oso xehea da eta konportamendu ona du egoera gehienetan
* Algoritmo honen helburua, array baten barruan ordenatuta dagoen veste array bat egitean datza
* While-ko bukle bakoitzean 0-tik a bukaeraraino ondo dagoen konprobatzen du
  + - 1. Algoritmoaren analisia
* Estabilitatea: Algoritmo honek ez du inoiz balore berdineko registroak elkartrukatzen, beraz, algoritmo establea da
* Memoria eskakizunak: Beste método askotan bezala, aldagai auxiliar baten beharra dago.
* Exekuzio denbora: Kanpoko buklea n - 1 aldiz egin behar. Barneko bukleak eginiko bueltak kanpoko buklearen araberakoak dira (kanpoko buklearen vuelta zenbakiaren veste vuelta ematen ditu barnekoan)
  + - 1. Kodigoa

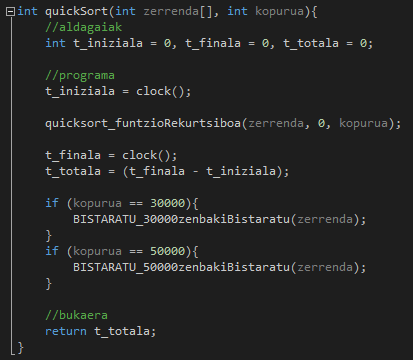


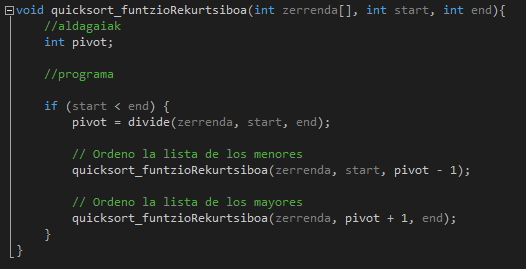
* + - 1. Abantailak
* Inplementazio erraza
* Memoria eskakizun txikiak
  + - 1. Desabantailak
* Geldoa
* Konparazio kantitate handiak egiten ditu
  1. ShellSort metodoa
     + 1. Azalpena
* Txertaketa zuzenaren eboluzio mejoratua bezalakoa da.
* ShellSort metodoak elementuen arteko elkartrukeak tamaina handiko saltoekin egiten da, baina inkremento dekrezenteekin
* Horrela zenbakiak arinago gelditzen dira ordenatuta
  + - 1. Algoritmoaren analisia
      2. Kodigoa

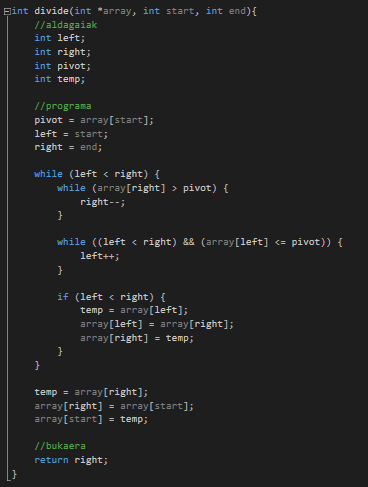


* + - 1. Abantailak
      2. Desabantailak
  1. QuickSort metodoa
     + 1. Azalpena
* Zerrenda guztitik elementu bat aukeratzen da
* Elementu horren ezkerraldean bera baino zenbaki txikiagoak jartzen dira eta eskuman bera baino handiagoak direnak
* 2 lista hoietan berriz berdina egiten da behin eta berriz, lista osoa ordenatu arte.
  + - 1. Algoritmoaren analisia
* Memoria eskakizunak: Funtzio errekurtsiboa izanez, okupatzen duen memoria zatia oso handia da.
* Exekuzio denbora: Buklerik ez duenez, bere exekuzio denbora oso txikia da.

* + - 1. Kodigoa







* + - 1. Abantailak
* Oso arina da
* Ez du memoria adizionalen beharrik
  + - 1. Desabantailak
* Inplementazio oso konplikatua
* Rekurtsibitatea erabiltzen duenez oso rekurtso asko erabiltzen ditu
* Diferentzia asko kasu hoberena eta txarrenaren artean

1. Metodoen exekuzio denboren konparaketa:

|  |  |  |
| --- | --- | --- |
| Metodo | 30 000 zenbaki | 50 000 zenbaki |
| Hautaketa zuzena | 2832 miliSeg | 2807 miliSeg |
| Hautaketa zuzena hobetua | 1366 miliSeg | 1346 miliSeg |
| Burbuila | 4314 miliSeg | 4368 miliSeg |
| Burbuila hobetua | 4193 miliSeg | 4163 miliSeg |
| Txertaketa zuzena | 1789 miliSeg | 1752 miliSeg |
| Txertaketa zuzena hobetua | 1759 miliSeg | 1830 miliSeg |
| ShellSort | 8 miliSeg | 8 miliSeg |
| QuickSort | 5 miliSeg | 5 miliSeg |

1. Metodoen konparaketa globala:

Metodo guztiak konparatuz, argi ikusten da azken biak (ShellSort eta QuickSort) denbora aldetik azkoz ere bizkorragoak direla. Erabiltzen duten memoriaren arabera, QuickSort-ak kenduta (memoria asko erabiltzen du errekurtsiboa izateagatik) beste guztiek antzerako memoria erabiltzen dute. Kodigoaren konplexutasuna kontuan izaten, QuickSort-ak kenduta (3 funtzio behar dira eta ez da erraza funtzio errekurtsiboa ulertzea) veste guztiak antzeko kodigo tamaina eta denak dira nahiko errazak ulertzeko. Beraz, dena kontuan artuta, metodorik onena ShellSort-a dela esango nuke, bere baino arinagoa den bakarra (QuickSort) oso konplexua delako, bera nahiko simplea izanik.