En el siguiente documento veremos cómo trabajar con los algoritmos de ordenamiento, así como: inserción, selección, bubble y Quicksort hablaremos de cómo funciona y sus códigos.

**Inserción:**

*El algoritmo se trabaja de la siguiente forma:*

Inicialmente se tiene un solo elemento, que obviamente es un conjunto ordenado. Después, cuando hay k elementos ordenados de menor a mayor, se toma el elemento k+1 y se compara con todos los elementos ya ordenados, deteniéndose cuando se encuentra un elemento menor (todos los elementos mayores han sido desplazados una posición a la derecha) o cuando ya no se encuentran elementos (todos los elementos fueron desplazados y este es el más pequeño). En este punto se inserta el elemento k+1 debiendo desplazarse los demás elementos.

*Su código es el siguiente:*

def ordenamiento\_insersion(A):  
 global cnt  
 for i in range (1,len(A)):  
 cnt+=1  
 valor= A[i]  
 q= i-1  
 while q >= 0:  
 if valor < A[q]:  
 A[q+1] = A[q]  
 A[q] = valor  
 q = q-1  
 else:  
 break

**Quicksort:**

*El algoritmo se trabaja de la siguiente forma:*

Elegimos un elemento de la lista ordenar al que llamaremos pivote, movemos cada elemento de la lista a cada lado del pivote los menores de un lado y los mayores del otro, la lista queda dividida en dos sublistas, se repite este procedo en cada sublista mientras contenga más de un elemento. Una vez terminado la lista estará ordenada.

*Su código es el siguiente:*

def ordenamiento\_quicksort(A):  
 global qc  
 if len(A) < 2:  
 return A  
 p= A.pop(0)  
 menores, mayores= list(), list()  
 for c in A:  
 qc+=1  
 if c <=p:  
 menores.append(c)  
 elif c > p:  
 mayores.append(c)  
 return ordenamiento\_quicksort(menores) + [p] + ordenamiento\_quicksort(mayores)

import random  
import  
  
def rndar (lon):  
 arr = []  
 for r in range(lon):  
 arr.append(random.randint(0,lon))  
 return arr

**Bubble:**

*El algoritmo se trabaja de la siguiente forma:*

Ir comparando desde la casilla 0 o desde cualquier otra casilla número tras número hasta encontrar uno mayor, si este es realmente el mayor de todo el vector se llevará hasta la última casilla, si no es así, será reemplazado por uno mayor que él. Este procedimiento seguirá así hasta que haya ordenado todas las casillas del vector.

*Su código es el siguiente:*

import random  
 import copy  
 def bubbleSort(alist):  
 contador = 0  
 for passnum in range(len(alist)-1,0,-1):  
 for i in range(passnum):  
 contador +=1  
 if alist[i]>alist[i+1]:  
 temp = alist[i]  
 alist[i] = alist[i+1]  
 alist[i+1] = temp  
 return contador

**Selection**

*El algoritmo se trabaja de la siguiente forma:*

Este algoritmo necesita n2 movimientos para ordenar una lista de números, toma el primer número y lo comparara con lo demás y al encontrar uno más pequeño lo toma y lo compara con el resto, ya cuando compara los todos los numero toma el menos y lo pone al principio y repite el proceso hasta ordenar todos los numero

*Su código es el siguiente:*

def selection(arr)  
 for i in range(0, len(arr)):  
 if arr[j]<arr[val]:  
 val=j   
 if yal !=[i]  
 aux=arr[i]  
 arr[i]=arr[val]  
 arr[val]= aux  
 return arr

***Grafica en R***

r= read.csv("E:/Evelyn/datos.txt",sep=" ")

> r

L B S I Q

1 4 6 6 7 5

2 4 6 6 6 5

3 4 6 6 6 4

4 4 6 6 8 6

5 4 6 6 7 5

6 4 6 6 7 4

7 4 6 6 7 5

8 4 6 6 7 4

9 4 6 6 5 6

10 4 6 6 8 6

11 4 6 6 5 4

12 4 6 6 7 6

13 4 6 6 3 6

14 4 6 6 3 4

15 4 6 6 5 4

16 4 6 6 7 4

17 4 6 6 6 4

18 4 6 6 4 4

19 4 6 6 4 6

20 4 6 6 6 6

21 4 6 6 3 6

22 4 6 6 8 4

23 4 6 6 7 6

24 4 6 6 6 6

25 4 6 6 4 5

26 4 6 6 4 5

27 4 6 6 6 4

28 4 6 6 5 6

29 4 6 6 6 6

30 4 6 6 9 6

31 8 28 28 25 18

32 8 28 28 9 24

33 8 28 28 20 22

34 8 28 28 22 14

35 8 28 28 22 18

36 8 28 28 17 15

37 8 28 28 18 24

38 8 28 28 23 18

39 8 28 28 20 16

40 8 28 28 23 13

41 8 28 28 19 18

42 8 28 28 19 14

43 8 28 28 16 19

44 8 28 28 22 14

45 8 28 28 16 13

46 8 28 28 14 16

47 8 28 28 18 20

48 8 28 28 24 16

49 8 28 28 18 20

50 8 28 28 12 16

51 8 28 28 19 16

52 8 28 28 27 14

53 8 28 28 24 21

54 8 28 28 19 20

55 8 28 28 15 14

56 8 28 28 25 14

57 8 28 28 13 15

58 8 28 28 20 15

59 8 28 28 19 17

60 8 28 28 17 14

61 16 120 120 97 56

62 16 120 120 64 46

63 16 120 120 79 45

64 16 120 120 67 48

65 16 120 120 82 49

66 16 120 120 85 62

67 16 120 120 72 58

68 16 120 120 78 51

69 16 120 120 69 42

70 16 120 120 94 56

71 16 120 120 72 45

72 16 120 120 71 49

73 16 120 120 67 50

74 16 120 120 82 51

75 16 120 120 64 42

76 16 120 120 93 68

77 16 120 120 89 59

78 16 120 120 75 56

79 16 120 120 65 56

80 16 120 120 76 49

81 16 120 120 64 41

82 16 120 120 69 62

83 16 120 120 76 54

84 16 120 120 73 47

85 16 120 120 57 82

86 16 120 120 79 66

87 16 120 120 83 83

88 16 120 120 69 50

89 16 120 120 103 46

90 16 120 120 72 47

91 32 496 496 301 125

92 32 496 496 331 127

93 32 496 496 302 135

94 32 496 496 311 127

95 32 496 496 244 118

96 32 496 496 309 163

97 32 496 496 270 142

98 32 496 496 256 151

99 32 496 496 250 135

100 32 496 496 226 139

101 32 496 496 278 131

102 32 496 496 251 123

103 32 496 496 226 147

104 32 496 496 263 136

105 32 496 496 288 143

106 32 496 496 274 122

107 32 496 496 229 138

108 32 496 496 250 143

109 32 496 496 289 151

110 32 496 496 270 137

111 32 496 496 292 168

112 32 496 496 291 139

113 32 496 496 303 139

114 32 496 496 274 179

115 32 496 496 252 156

116 32 496 496 273 132

117 32 496 496 275 147

118 32 496 496 256 125

119 32 496 496 290 131

120 32 496 496 247 163

121 64 2016 2016 993 396

122 64 2016 2016 877 345

123 64 2016 2016 1014 376

124 64 2016 2016 1062 371

125 64 2016 2016 978 387

126 64 2016 2016 1010 353

127 64 2016 2016 928 364

128 64 2016 2016 1094 337

129 64 2016 2016 1116 345

130 64 2016 2016 1032 328

131 64 2016 2016 1084 342

132 64 2016 2016 1022 408

133 64 2016 2016 1033 389

134 64 2016 2016 940 317

135 64 2016 2016 1023 325

136 64 2016 2016 1060 342

137 64 2016 2016 1115 409

138 64 2016 2016 1156 443

139 64 2016 2016 955 345

140 64 2016 2016 1041 364

141 64 2016 2016 941 305

142 64 2016 2016 987 313

143 64 2016 2016 976 365

144 64 2016 2016 1153 368

145 64 2016 2016 1238 378

146 64 2016 2016 1076 329

147 64 2016 2016 1053 337

148 64 2016 2016 1029 381

149 64 2016 2016 1196 332

150 64 2016 2016 974 343

151 128 8128 8128 3829 844

152 128 8128 8128 4750 846

153 128 8128 8128 4143 829

154 128 8128 8128 4288 801

155 128 8128 8128 3740 805

156 128 8128 8128 4131 808

157 128 8128 8128 4110 814

158 128 8128 8128 4255 1003

159 128 8128 8128 4035 980

160 128 8128 8128 4178 1003

161 128 8128 8128 4161 876

162 128 8128 8128 3817 811

163 128 8128 8128 3954 961

164 128 8128 8128 3982 827

165 128 8128 8128 4127 865

166 128 8128 8128 3855 873

167 128 8128 8128 3980 869

168 128 8128 8128 4137 888

169 128 8128 8128 4459 875

170 128 8128 8128 4034 862

171 128 8128 8128 4264 850

172 128 8128 8128 4059 853

173 128 8128 8128 4440 855

174 128 8128 8128 3965 777

175 128 8128 8128 4162 974

176 128 8128 8128 3584 901

177 128 8128 8128 3952 1047

178 128 8128 8128 3558 881

179 128 8128 8128 4258 811

180 128 8128 8128 4341 892

181 256 32640 32640 17341 2225

182 256 32640 32640 16017 1891

183 256 32640 32640 15862 2143

184 256 32640 32640 16401 2051

185 256 32640 32640 16683 2122

186 256 32640 32640 16212 1993

187 256 32640 32640 16005 2385

188 256 32640 32640 15981 1975

189 256 32640 32640 16723 2133

190 256 32640 32640 16842 1980

191 256 32640 32640 17419 2622

192 256 32640 32640 16427 1950

193 256 32640 32640 16265 2078

194 256 32640 32640 17245 1971

195 256 32640 32640 15608 2025

196 256 32640 32640 16511 2132

197 256 32640 32640 16601 2037

198 256 32640 32640 15858 2293

199 256 32640 32640 16703 1996

200 256 32640 32640 16713 2500

201 256 32640 32640 15760 2293

202 256 32640 32640 16934 2092

203 256 32640 32640 15301 2207

204 256 32640 32640 16182 2092

205 256 32640 32640 17957 2274

206 256 32640 32640 16847 2003

207 256 32640 32640 15741 2269

208 256 32640 32640 15869 2255

209 256 32640 32640 16170 2121

210 256 32640 32640 15433 2033

211 512 130816 130816 66664 4871

212 512 130816 130816 66781 4961

213 512 130816 130816 66246 4840

214 512 130816 130816 61988 5065

215 512 130816 130816 65999 4641

216 512 130816 130816 64833 4749

217 512 130816 130816 63486 6220

218 512 130816 130816 64243 5057

219 512 130816 130816 64715 4839

220 512 130816 130816 66083 5215

221 512 130816 130816 64368 4938

222 512 130816 130816 66986 5493

223 512 130816 130816 67014 4722

224 512 130816 130816 64776 5274

225 512 130816 130816 65366 5057

226 512 130816 130816 65642 5137

227 512 130816 130816 67297 5099

228 512 130816 130816 61873 4546

229 512 130816 130816 65705 5109

230 512 130816 130816 64184 5146

231 512 130816 130816 63543 4765

232 512 130816 130816 66709 5250

233 512 130816 130816 65480 4390

234 512 130816 130816 64304 4870

235 512 130816 130816 67518 6085

236 512 130816 130816 64282 4647

237 512 130816 130816 65674 4879

238 512 130816 130816 65947 5238

239 512 130816 130816 66022 4909

240 512 130816 130816 64631 5355

241 1024 523776 523776 258324 11109

242 1024 523776 523776 263552 10941

243 1024 523776 523776 266602 10894

244 1024 523776 523776 265516 11718

245 1024 523776 523776 265256 11054

246 1024 523776 523776 261984 10412

247 1024 523776 523776 272710 10955

248 1024 523776 523776 260328 11346

249 1024 523776 523776 263215 11287

250 1024 523776 523776 265879 11234

251 1024 523776 523776 268092 10399

252 1024 523776 523776 262485 10341

253 1024 523776 523776 262341 10747

254 1024 523776 523776 277197 10802

255 1024 523776 523776 260699 11080

256 1024 523776 523776 267757 11060

257 1024 523776 523776 256225 11067

258 1024 523776 523776 262735 12727

259 1024 523776 523776 254194 11819

260 1024 523776 523776 259252 10762

261 1024 523776 523776 253405 11310

262 1024 523776 523776 262245 10688

263 1024 523776 523776 266106 10667

264 1024 523776 523776 263650 11417

265 1024 523776 523776 261735 11196

266 1024 523776 523776 254066 11240

267 1024 523776 523776 257378 11136

268 1024 523776 523776 253505 11868

269 1024 523776 523776 267382 11579

270 1024 523776 523776 262911 11097

> plot(r$I~r$L)

> plot(r$Q~r$L, type='l')

> lines(r$B~r$L, col='red')

> lines(r$I~r$L, col='green')

> lines(r$S~r$L, col='blue')

