1 **/\***

2 **\* Ovo je prijepis mog hibridnog algoritma razvrstavanja da se može**

3 **\* pokrenuti na JavaScriptinoj virtualnoj mašini. Kako bi se AEC-om mogla**

4 **\* ciljati JavaScriptina virtualna mašina, napravio sam novi compiler.**

5 **\* Ovaj puta je pisan u C++-u, radio sam i novi parser i novi tokenizer.**

6 **\* Također sam malo promijenio sintaksu, da omogućim pisanje čistijeg koda**

7 **\* te da podržavam različite vrste podataka (prije je AEC podržavao samo**

8 **\* 32-bitne decimalne brojeve). Novi compiler proizvodi WebAssembly,**

9 **\* standardizirani oblik JavaScriptinog bytecodea. Izvorno je WebAssembly**

10 **\* bio Mozillin standard, ali danas ga podržavaju gotovo sve JavaScriptine**

11 **\* virtualne mašine. Ovdje ciljamo primarno na NodeJS, JavaScriptinu**

12 **\* virtualnu mašinu koju razvija Google i primarno je namijenjena da se**

13 **\* vrti na serverima (no može se pokrenuti i na veoma slabim računalima).**

14 **\*/**

15

16 **// Uvezimo prvo neke funkcije iz JavaScripta, koje će nam trebati...**

17 **Function** daj\_velicinu\_niza**()** **Which** **Returns** **Integer32** **Is** **External;**

18 **Function** kopiraj\_niz\_na\_adresu**(**

19 **Integer32Pointer** adresa**)** **Which** **Returns** **Nothing** **Is** **External;**

20 **Function** printString**(CharacterPointer** str**)** **Which** **Returns** **Nothing** **Is** **External;**

21 **Function** printInteger**(Integer32** n**)** **Which** **Returns** **Nothing** **Is** **External;**

22 **Function** printFloat**(Decimal32** x**)** **Which** **Returns** **Nothing** **Is** **External;**

23 **Function** pocni\_mjerenje\_vremena**()** **Which** **Returns** **Nothing** **Is** **External;**

24 **Function** zavrsi\_mjerenje\_vremena**()** **Which** **Returns** **Nothing** **Is** **External;**

25 **Function** izvijesti\_o\_obrnuto\_poredanim\_nizovima**(**

26 **Integer32** n**)** **Which** **Returns** **Nothing** **Is** **External;**

27 **Function**

28 izvijesti\_o\_poredanim\_nizovima**(Integer32** n**)** **Which** **Returns** **Nothing** **Is** **External;**

29 **Function** izvijesti\_o\_pokretanju\_QuickSorta**(**

30 **Integer32** n**)** **Which** **Returns** **Nothing** **Is** **External;**

31 **Function** izvijesti\_o\_pokretanju\_MergeSorta**(**

32 **Integer32** n**)** **Which** **Returns** **Nothing** **Is** **External;**

33 **Function** izvijesti\_o\_pokretanju\_SelectSorta**(**

34 **Integer32** n**)** **Which** **Returns** **Nothing** **Is** **External;**

35 **Function**

36 izvijesti\_JavaScript\_o\_nedostatku\_memorije**()** **Which** **Returns** **Nothing** **Is** **External;**

37

38 **Integer32** DEBUG **:** **=** **0,** broj\_mjerenja **:** **=** **0;** **// broj\_mjerenja bit će koristan za**

39 **// pokretanje raznih algoritama**

40 **// ovisno o rednom broju mjerenja,**

41 **//što dobro dođe za**

42 **// eksperimentiranje s algoritmima.**

43

44 **// Napravimo sada omotnicu oko WebAssemblerske naredbe "memory.grow"...**

45 **Function** zauzmi\_memorijske\_stranice**(Integer32** broj\_stranica**)** **Which** **Returns**

46 **CharacterPointer**

47 **Does** **{** **// Vitičasta zagrada ovdje je opcionalna, compiler ju ignorira. No,**

48 **// dobro dođe kada pišemo AEC-ovski program u nekom IDE-u koji je**

49 **// primarno namijenjen za C-olike jezike, tada on može razumijeti da**

50 **// je ono između vitičastih zagrada blok naredbi.**

51 **Integer32** nova\_adresa\_u\_stranicama

52 **:** **=** **asm\_i32** **//"asm\_i32" kaže compileru da umetne asemblerski kod, i da**

53 **// pretpostavi da će se nakon njega na sistemskom stogu**

54 **// nalaziti vrijednost tipa "i32". To očito nije točno ako**

55 **// netko prebaci JavaScript virtualnu mašinu u 64-bitni**

56 **// način rada, ali nadam se da to nitko neće napraviti.**

57 **// Vjerojatnost da će JavaScript virtualnoj mašini trebati**

58 **// više nego 4GB RAM-a je zanemariva, a vjerojatnost da će**

59 **// se neki korisni programi srušiti zbog prebacivanja u**

60 **// 64-bitni način rada nije baš zanemariva.**

61 **(**"(memory.grow**\*n***"

62 "**\*t***(local.get 0)**\*n***" **// Prvi (nulti) argument funkcije,**

63 **//"broj\_stranica".**

64 ")**\*n***"**);**

65 **If** nova\_adresa\_u\_stranicama **=** **-1** **Then** **{** **// Ako nema više**

66 **// slobodne memorije...**

67 **Return(-1);** **// Ne stavim li zagrade oko "-1", ClangFormat će misliti da je**

68 **// "Return" ime neke varijable.**

69 **}**

70 **EndIf;** **// Ni točka-zarez ovdje nije potreban, ali pomaže IDE-ovima da razumiju**

71 **// programski kôd.**

72 **Return** nova\_adresa\_u\_stranicama **\*** **64** **\*** **1024;** **// Na JavaScript Virtualnoj**

73 **// Mašini, jedna stranica**

74 **//(page) iznosi 64 KB.**

75 **}**

76 **EndFunction;**

77

78 **Integer32** velicina\_niza**;**

79 **Integer32Pointer** originalni\_niz**,** pomocni\_niz**;** **// To su globalne varijable,**

80 **// po defaultu su u nuli,**

81 **// dakle "originalni\_niz" i**

82 **//"pomocni\_niz" su na početku**

83 **// programa nulti pokazivači.**

84

85 **Integer32** broj\_obrnuto\_poredanih\_podniza**,** broj\_vec\_poredanih\_podniza**,**

86 broj\_pokretanja\_QuickSorta**,** broj\_pokretanja\_MergeSorta**,**

87 broj\_pokretanja\_SelectSorta**;**

88

89 **// Sad ćemo implementirati neke matematičke funkcije koje će nam trebati.**

90 **// Ne možemo pozvati JavaScriptine matematičke funkcije, jer one su metode**

91 **// singletona "Math", a ne postoji standardizirani način da se zovu**

92 **// metode JavaScriptinih objekata iz WebAssemblyja.**

93 **Decimal32** PRECISION **:** **=** **128;** **// Ovdje možemo balansirati između brzine i**

94 **// preciznosti. Ako smo previše precizni, bit**

95 **//ćemo spori. Ako smo previše neprecizni, lako**

96 **// se može dogoditi da precijenimo koliko se**

97 **// duboko rekurzija smije granati i izazovemo**

98 **// stack overflow.**

99

100 **Decimal32**

101 memoizacija\_prirodnog\_logaritma**[8192];** **// Memoizacija je dio memorije u koju**

102 **// se spremaju već izračunati podaci,**

103 **// da se ne računaju više puta.**

104

105 **Function** Eulerov\_broj\_na\_potenciju**(Decimal32** x**)**

106 **Which** **Returns** **Decimal32** **Is** **Declared;** **// Vidi dolje za pojašnjenje ovoga.**

107

108 **Function** prirodni\_logaritam**(Decimal32** x**)** **Which** **Returns** **Decimal32** **Does** **{**

109 **If** x **<** **1** **Then** **{** **Return(-**prirodni\_logaritam**(1** **/** x**));** **}**

110 **EndIf;**

111 **If(**x **<** **8192** **and**

112 **not(**memoizacija\_prirodnog\_logaritma**[asm\_f32(** **// Vidi iduću funkciju za**

113 **// pojašnjenje ovoga...**

114 "(f32.nearest (local.get 0))"**)]** **=** **0))** **Then** **{**

115 **Return**

116 memoizacija\_prirodnog\_logaritma**[asm\_f32(**"(f32.nearest (local.get 0))"**)];**

117 **}**

118 **EndIf;**

119 **If** x **>** **8192** **Then** **{**

120 **Return** **1** **+** prirodni\_logaritam**(**x **/** Eulerov\_broj\_na\_potenciju**(1));**

121 **}**

122 **EndIf;**

123 **// Prirodni logaritam je integral od 1/x u intervalu od 1 do x,**

124 **// srednjoškolska matematika.**

125 **Decimal32** zbroj **:** **=** **0,** epsilon **:** **=** **(**x **-** **1)** **/** **(5** **\*** PRECISION**),** i **:** **=** **1;**

126 **While(**epsilon **>** **0** **and** i **<** x**)** **or** **(**epsilon **<** **0** **and** i **>** x**)** **Loop** **{**

127 zbroj **+=** epsilon **/** i**;**

128 i **+=** epsilon**;** **// Preuzeo sam naredbe "+=", "-=", "\*=" i "/=" iz C-olikih**

129 **// jezika, smatram da znatno skraćuju neke kodove, a da ih ne**

130 **// čine nečitkima.**

131 **}**

132 **EndWhile;**

133 **If** x **<** **8192** **Then** **{**

134 **Return**

135 memoizacija\_prirodnog\_logaritma**[asm\_f32(**"(f32.nearest (local.get 0))"**)]**

136 **:** **=** zbroj**;**

137 **}**

138 **EndIf;**

139 **Return** zbroj**;**

140 **}**

141 **EndFunction;**

142

143 **Decimal32**

144 memoizacija\_Eulerovog\_algoritma**[256];** **// Možda malo ubrza program, gotovo**

145 **// sigurno ga neće usporiti...**

146

147 **Function** abs**(Decimal32** x**)** **Which** **Returns** **Decimal32** **Is**

148 **Declared;** **// Deklarirat ćemo funkciju "abs" da se može koristiti prije no**

149 **// što se definira.**

150

151 **Function** Eulerov\_broj\_na\_potenciju**(Decimal32** x**)** **Which** **Returns** **Decimal32** **Does** **{**

152 **If(**abs**(**x**)** **<** **64** **and**

153 **not(**memoizacija\_Eulerovog\_algoritma

154 **[asm\_f32(**"(f32.nearest (local.get 0))"**)** **\*** **2** **+** **128]** **=** **0))**

155 **Then** **{** **// Koristio sam ovdje umetnuti asembler zato što ništa drugo što**

156 **// napišem u AEC-u neće učiniti da compiler koji sam napravio**

157 **// pozove asemblersku instrukciju "f32.nearest" (poznatu kao**

158 **// "round" u drugim programskim jezicima), a mnogo je lakše**

159 **// napisati malo umetnutog asemblera nego mijenjati i recompilirati**

160 **// compiler.**

161 **Return** memoizacija\_Eulerovog\_algoritma

162 **[asm\_f32(**"(f32.nearest (local.get 0))"**)** **\*** **2** **+** **128];**

163 **}**

164 **EndIf;**

165 **// Eulerov Algoritam iz Matematike 2...**

166 **Decimal32** i **:** **=** **0,** y **:** **=** **1,** epsilon **:** **=** x **/** PRECISION**;**

167 **While(**epsilon **>** **0** **and** i **<** x**)** **or** **(**epsilon **<** **0** **and** i **>** x**)** **Loop** **{**

168 y **+=** epsilon **\*** y**;**

169 i **+=** epsilon**;**

170 **}**

171 **EndWhile;**

172 **If** abs**(**x**)** **<** **64** **Then** **{**

173 **Return** memoizacija\_Eulerovog\_algoritma

174 **[asm\_f32(**"(f32.nearest (local.get 0))"**)** **\*** **2** **+** **128]** **:** **=** y**;**

175 **}**

176 **EndIf** **Return** y**;**

177 **}**

178 **EndFunction;**

179

180 **Function** abs**(Decimal32** x**)** **Which** **Returns** **Decimal32** **Does** **{**

181 **// U svoj sam programski jezik ugradio uvijetni "?:" operator kakav**

182 **// postoji u C-u, C++-u i JavaScriptu. Izgleda malo ružno, ali nekad zna**

183 **// znatno skratiti programske kodove. Odlučio sam implementirati desno**

184 **// asocijativan uvijetni operator, kakav je u C-u, C++-u i JavaScriptu,**

185 **// a ne lijevo asocijativan kakav je u PHP-u i srodnim jezicima.**

186 **// Jednostavno mi ima više smisla da uvijetni operator bude asocijativan**

187 **// na desno nego na lijevo.**

188 **Return(**x **<** **0)** **?** **// Ako je x manji od 0...**

189 **-**x **//...vrati (proglasi rezultatom) -x...**

190 **:** x**;** **// inače, proglasi x rezultatom.**

191 **}**

192 **EndFunction;**

193

194 **Function** ostatak\_pri\_dijeljenju**(Decimal32** x**,**

195 **Decimal32** y**)** **Which** **Returns** **Decimal32** **Does** **{**

196 **If** DEBUG **=** **1** **Then** **{**

197 printString**(**"Zatrazen je ostatak pri dijeljenju od brojeva: "**);**

198 **// Neću upotrebljavati hrvatske znakove u stringovima, jer ću**

199 **// naletjeti na probleme pri pretvorbi u JavaScriptin string.**

200 printFloat**(**x**);**

201 printFloat**(**y**);**

202 printString**(**"Sada ce se program mozda srusiti..."**);**

203 **}**

204 **EndIf;**

205 **If** abs**(**x **/** y**)** **>** Eulerov\_broj\_na\_potenciju**(**prirodni\_logaritam**(2)** **\*** **63)** **Then** **{**

206 **Return** **0;** **// Imate bolju ideju što da se radi u slučaju da količnik**

207 **// ne stane niti u Integer64 (C-ovski "long long")?**

208 **}**

209 **EndIf;**

210 **Return** x **-** y **\*Integer64(**x **/** y**);** **// Ako napišem "Integer32",**

211 **// riskiram da će JavaScript**

212 **// virtualna mašina prekinuti**

213 **// izvođenje programa jer je**

214 **// broj "x/y" izvan intervala**

215 **// koji 32-bitni cijeli brojevi**

216 **// mogu prikazati (od oko dvije**

217 **// milijarde u pozitivno i**

218 **// negativno).**

219 **}**

220 **EndFunction;**

221

222 **Function** pow**(Decimal32** x**,** **Decimal32** y**)** **Which** **Returns** **Decimal32** **Does** **{**

223 **Decimal32** result

224 **:** **=** Eulerov\_broj\_na\_potenciju**(**prirodni\_logaritam**(**abs**(**x**))** **\*** y**);**

225 **Return** x **=**

226 **0** **?** **0** **:** ostatak\_pri\_dijeljenju**(**x**,** **2)** **=** **1** **and** x **<** **0** **?** **-**result **:** result**;**

227 **}**

228 **EndFunction;**

229

230 **// I sada krećemo pisati taj hibridni algoritam razvrstavanja...**

231 **Function** hybrid\_sort**(Integer32** donja\_granica**,** **Integer32** gornja\_granica**,**

232 **Integer32** dubina\_rekurzije**)** **Which** **Returns** **Nothing** **Does** **{**

233 **If** gornja\_granica **-** donja\_granica **<** **2** **Then** **{** **// Ako je niz duljine manje od**

234 **// 2 (0 ili 1), znači da je već**

235 **// poredan, pa prekidamo**

236 **// izvođenje ovog potprograma.**

237 **Return;**

238 **}**

239 **ElseIf** gornja\_granica **-** donja\_granica **=** **2** **Then** **{** **// Najčesći slučaj,**

240 **// vrijedi ga posebno**

241 **// obraditi jer time**

242 **// možemo znatno ubrzati**

243 **// program.**

244 **If** **ValueAt(**originalni\_niz **+** donja\_granica**)** **>**

245 **ValueAt(**originalni\_niz **+** donja\_granica **+** **1)** **Then** **{**

246 **Integer32** pomocna\_varijabla\_za\_zamijenu

247 **:** **=** **ValueAt(**originalni\_niz **+** donja\_granica**);**

248 **ValueAt(**originalni\_niz **+** donja\_granica**)**

249 **:** **=** **ValueAt(**originalni\_niz **+** donja\_granica **+** **1);**

250 **ValueAt(**originalni\_niz **+** donja\_granica **+** **1)**

251 **:** **=** pomocna\_varijabla\_za\_zamijenu**;**

252 **}**

253 **EndIf;**

254 **Return;**

255 **}**

256 **ElseIf** gornja\_granica **-**

257 donja\_granica**<8** **or** **asm\_i32(**"(global.get $stack\_pointer)"**)>** **4** **\*** **1024** **-**

258 **73** **Then** **{**

259 **// Za male je nizove SelectionSort brži i od MergeSorta i QuickSorta. Također,**

260 **// kako nije rekurzivan, može se koristiti i kad posve potrošimo memoriju na**

261 **// sistemskom stogu(na JavaScript Virtualnoj Mašini to jest ne više nego 4 KB,**

262 **// kako bijaše u doba Netscapea 2, godine 1996, tako i danas).**

263 broj\_pokretanja\_SelectSorta**:**

264 **=** broj\_pokretanja\_SelectSorta **+** **1;**

265 **Integer32** i **:** **=** donja\_granica**;**

266 **While** i **<** gornja\_granica **Loop** **{**

267 **Integer32** gdje\_je\_minimum **:** **=** i**;**

268 **Integer32** j **:** **=** i **+** **1;**

269 **While** j **<** gornja\_granica **Loop** **{**

270 **If** **ValueAt(**originalni\_niz **+** gdje\_je\_minimum**)** **>**

271 **ValueAt(**originalni\_niz **+** j**)** **Then** **{**

272 gdje\_je\_minimum

273 **/\***

274 **\* ClangFormat (koji koristim za formatiranje AEC programa, a**

275 **\* primarno je namijenjen za C-olike jezike) pogrešno tumači**

276 **\* AEC-ov operator pridruživanja ":=" kao C-ovu oznaku za labele**

277 **\* ':' plus C-ov operator pridruživanja '='. Ne vidim nekakvo**

278 **\* jednostavno rješenje tog problema. Na sreću, AEC-ov tokenizer,**

279 **\* još od najranije verzije, trpi ako se stavi whitespace znak**

280 **\* (razmak, tabulator ili znak za novi red) između ':' i '=' u**

281 **\* operatoru ":=", ali svejedno to ne izgleda dobro.**

282 **\*/**

283 **:**

284 **=** j**;**

285 **}**

286 **EndIf;**

287 j **+=** **1;**

288 **}**

289 **EndWhile;**

290 **Integer32** pomocna\_varijabla\_za\_zamijenu **:** **=** **ValueAt(**originalni\_niz **+** i**);**

291 **ValueAt(**originalni\_niz **+** i**)** **:** **=** **ValueAt(**originalni\_niz **+** gdje\_je\_minimum**);**

292 **ValueAt(**originalni\_niz **+** gdje\_je\_minimum**)**

293 **:** **=** pomocna\_varijabla\_za\_zamijenu**;**

294 i **+=** **1;**

295 **}**

296 **EndWhile;**

297 **Return;**

298 **}**

299 **EndIf;**

300 **Decimal32** razvrstanost **:** **=** **0;**

301 **Integer32** i **:** **=** donja\_granica**,** je\_li\_niz\_vec\_poredan **:** **=** **1;**

302 **While** i **<** gornja\_granica **-** **1** **Loop** **{**

303 razvrstanost **+=**

304 **(ValueAt(**originalni\_niz **+** i**)** **<** **ValueAt(**originalni\_niz **+** i **+** **1)** **or**

305 **ValueAt(**originalni\_niz **+** i**)** **=** **ValueAt(**originalni\_niz **+** i **+** **1));**

306 **If** **ValueAt(**originalni\_niz **+** i**)** **>** **ValueAt(**originalni\_niz **+** i **+** **1)** **Then** **{**

307 je\_li\_niz\_vec\_poredan**:**

308 **=** **0;**

309 **}**

310 **EndIf;**

311 i **+=** **1;**

312 **}**

313 **EndWhile;**

314 razvrstanost**:**

315 **=** razvrstanost **/** **((**gornja\_granica **-** donja\_granica **-** **1)** **/** **2.)** **-** **1;**

316 **// Provjeri je li sve u redu, i, ako nije, obavijesti.**

317 **If** abs**(**razvrstanost**)** **>** **1** **Then** **{**

318 **// To ne smije biti...**

319 printString**(**"Apsolutna vrijednost razvrstanosti je veca od 1!"**);**

320 printString**(**"Relevantni dio niza iznosi:"**);** **// Da se ne moram baktati s**

321 **// debuggerima za JavaScript**

322 **// virtualnu mašinu ako dođe**

323 **// do problema, lakše mi**

324 **// je ispisati brojeve u**

325 **// programu nego tražiti**

326 **// kako narediti**

327 **// debuggeru da ih**

328 **// ispiše.**

329 i**:**

330 **=** donja\_granica**;**

331 **While** i **<** gornja\_granica **Loop** **{**

332 printInteger**(ValueAt(**originalni\_niz **+** i**));**

333 i **+=** **1;**

334 **}**

335 **EndWhile;**

336 printString**(**"Kraj relevantnog dijela niza!"**);**

337 **}**

338 **EndIf;**

339 **If** je\_li\_niz\_vec\_poredan **and** **not(**razvrstanost **=** **1** **or** razvrstanost **>** **1)** **Then** **{**

340 **// Opet ne smije biti...**

341 printString**(**"Niz je poredan, a razvrstanost nije 1."**);**

342 printString**(**"Relevantni dio niza iznosi:"**);**

343 i**:**

344 **=** donja\_granica**;**

345 **While** i **<** gornja\_granica **Loop** **{**

346 printInteger**(ValueAt(**originalni\_niz **+** i**));**

347 i **+=** **1;**

348 **}**

349 **EndWhile;**

350 printString**(**"Kraj relevantnog dijela niza!"**);**

351 **}**

352 **EndIf;**

353 **If** **not(**je\_li\_niz\_vec\_poredan**)** **and** **(**razvrstanost **=** **1** **or** razvrstanost **>** **1)**

354 **Then** **{**

355 **// Open ne smije biti...**

356 printString**(**"Razvrstanost je 1, a niz nije poredan!"**);**

357 printString**(**"Relevantni dio niza iznosi:"**);**

358 i**:**

359 **=** donja\_granica**;**

360 **While** i **<** gornja\_granica **Loop** **{**

361 printInteger**(ValueAt(**originalni\_niz **+** i**));**

362 i **+=** **1;**

363 **}**

364 **EndWhile;**

365 printString**(**"Kraj relevantnog dijela niza!"**);**

366 **}**

367 **EndIf;**

368 **// Idemo dalje...**

369 **Decimal32** razvrstanost\_na\_potenciju**[8]** **:** **=** **{1};** **// Formula će se brže**

370 **// izračunati ako ne**

371 **// pozivamo "pow" gdje**

372 **// ne treba (kad je**

373 **// eksponent prirodan**

374 **// broj).**

375i**:**

376 **=** **1;**

377 **While** i **<** **8** **Loop** **{**

378 razvrstanost\_na\_potenciju**[**i**]**

379 **:** **=** razvrstanost\_na\_potenciju**[**i **-** **1]** **\*** razvrstanost**;**

380 i **+=** **1;**

381 **}**

382 **EndWhile;**

383 **// Formula koju je ispisao genetski algoritam za predviđanje koliko će**

384 **// usporedbi QuickSort napraviti:**

385 **// https://github.com/FlatAssembler/ArithmeticExpressionCompiler/tree/master/QuickSort/Genetic\_algorithm\_for\_deriving\_the\_formula**

386 **Decimal32** polinom\_pod\_apsolutnom

387 **:** **=** **2.38854** **\*** razvrstanost\_na\_potenciju**[7]** **-**

388 **0.284258** **\*** razvrstanost\_na\_potenciju**[6]** **-**

389 **1.87104** **\*** razvrstanost\_na\_potenciju**[5]** **+**

390 **0.372637** **\*** razvrstanost\_na\_potenciju**[4]** **+**

391 **0.167242** **\*** razvrstanost\_na\_potenciju**[3]** **-**

392 **0.0884977** **\*** razvrstanost\_na\_potenciju**[2]** **+** **0.315119** **\*** razvrstanost**;**

393 **Decimal32** Eulerov\_broj\_na\_koju\_potenciju

394 **:** **=** **(**prirodni\_logaritam**(**gornja\_granica **-** donja\_granica**)** **+**

395 prirodni\_logaritam**(**

396 prirodni\_logaritam**(**gornja\_granica **-** donja\_granica**)))** **\***

397 **1.05** **+**

398 **(**prirodni\_logaritam**(**gornja\_granica **-** donja\_granica**)** **-**

399 prirodni\_logaritam**(**

400 prirodni\_logaritam**(**gornja\_granica **-** donja\_granica**))** **-**

401 prirodni\_logaritam**(2))** **\***

402 **0.9163** **\*** abs**(**polinom\_pod\_apsolutnom**);**

403 **Decimal32** koliko\_usporedbi\_ocekujemo\_od\_QuickSorta

404 **:** **=** Eulerov\_broj\_na\_potenciju**(**Eulerov\_broj\_na\_koju\_potenciju**);**

405 **Decimal32** koliko\_usporedbi\_ocekujemo\_od\_MergeSorta

406 **:** **=** **(mod(**broj\_mjerenja**,** **2)** **+** **1)** **\***

407 **(**gornja\_granica **-**

408 donja\_granica**)** **\*** **// Nisam siguran treba li ovdje**

409 **// pisati "2 \* (gornja\_granica...".**

410 **// S jedne strane, MergeSort radi**

411 **// dvije petlje, jedna za spajanje**

412 **// dijelova originalnog niza u**

413 **// pomoćni niz, a druga za kopiranje**

414 **// pomoćnog niza u originalni. S**

415 **// druge strane, iz mjerenja se čini**

416 **// da je cjelokupni algoritam brži**

417 **// ako se ne množi s 2. Zato ćemo**

418 **// nekada množiti s dva, a nekada ne,**

419 **// pa ćemo preciznijim mjerenjima**

420 **// vidjeti što je bolje.**

421 prirodni\_logaritam**(**gornja\_granica **-** donja\_granica**)** **/**

422 prirodni\_logaritam**(2);**

423 **// I sada kreće grananje na temelju izračunatog...**

424 **If** razvrstanost **=** **1** **or** razvrstanost **>** **1** **Then** **{**

425 broj\_vec\_poredanih\_podniza **+=** **1;**

426 **Return;**

427 **}**

428 **ElseIf** razvrstanost **=** **-1** **or** razvrstanost **<** **-1** **Then** **{**

429 broj\_obrnuto\_poredanih\_podniza **+=** **1;**

430 **Integer32** i **:** **=** donja\_granica**;**

431 **Integer32** j **:** **=** gornja\_granica **-** **1;**

432 **While** i **<** gornja\_granica **Loop** **{**

433 **ValueAt(**pomocni\_niz **+** i**)** **:** **=** **ValueAt(**originalni\_niz **+** j**);**

434 j **-=** **1;**

435 i **+=** **1;**

436 **}**

437 **EndWhile;**

438 i**:**

439 **=** donja\_granica**;**

440 **While** i **<** gornja\_granica **Loop** **{**

441 **ValueAt(**originalni\_niz **+** i**)** **:** **=** **ValueAt(**pomocni\_niz **+** i**);**

442 i **+=** **1;**

443 **}**

444 **EndWhile;**

445 **Return;**

446 **}**

447 **ElseIf(**koliko\_usporedbi\_ocekujemo\_od\_MergeSorta **<**

448 koliko\_usporedbi\_ocekujemo\_od\_QuickSorta **or**

449 dubina\_rekurzije **>** pow**(2,** **18** **-** prirodni\_logaritam**(**velicina\_niza**)** **/**

450 prirodni\_logaritam**(2))**

451 **// JavaScriptina virtualna mašina ima**

452 **// 4KB memorije na sistemskom stogu,**

453 **// i alociranje više heap memorije**

454 **// ne mijenja tu nesretnu činjenicu.**

455 **// Ne znam kako Emscripten (modificirana**

456 **// verzija CLANG-a koja compilira**

457 **// C++ u WebAssembly) to rješava.**

458 **)** **and**

459 **not(**gornja\_granica **-** donja\_granica **=**

460 velicina\_niza **and** **not(mod(**broj\_mjerenja**,** **3)))**

461 **// Izgleda da je, iz nekog razloga, program brži ako se QuickSort**

462 **// pokrene barem jednom, no probajmo raditi preciznija mjerenja.**

463 **Then** **{**

464 **// MergeSort algoritam (približno poredani podnizovi,**

465 **// za koje je MergeSort efikasniji od QuickSorta,**

466 **// a moj ga program također koristi kada ima još**

467 **// malo mjesta na sistemskom stogu, pa QuickSort**

468 **// nije opcija)...**

469 broj\_pokretanja\_MergeSorta **+=** **1;**

470 **Integer32** sredina\_niza **:** **=** **(**gornja\_granica **+** donja\_granica**)** **/** **2;**

471 **// Prvo, rastavi niz na koji pokazuje pokazivač "originalni\_niz"**

472 **// na niz od originalni\_niz+donja\_granica do**

473 **// originalni\_niz+sredina\_niza i niz od**

474 **// originalni\_niz+sredina\_niza do**

475 **// originalni\_niz+gornja\_granica,**

476 **// i poredaj ta dva niza.**

477 hybrid\_sort**(**donja\_granica**,** sredina\_niza**,** dubina\_rekurzije **+** **1);**

478 hybrid\_sort**(**sredina\_niza**,** gornja\_granica**,** dubina\_rekurzije **+** **1);**

479 **// Spajanje nizova originalni\_niz[donja\_granica..sredina\_niza]**

480 **// i originalni\_niz[sredina\_niza..gornja\_granica] u jedan niz...**

481 **Integer32** i **:** **=** donja\_granica**;**

482 **Integer32** gdje\_smo\_u\_prvom\_nizu **:** **=** donja\_granica**;**

483 **Integer32** gdje\_smo\_u\_drugom\_nizu **:** **=** sredina\_niza**;**

484 **While** i **<** gornja\_granica **Loop** **{**

485 **If(**gdje\_smo\_u\_prvom\_nizu **=**

486 sredina\_niza **or**

487 **ValueAt(**originalni\_niz **+** gdje\_smo\_u\_drugom\_nizu**)** **<**

488 **ValueAt(**originalni\_niz **+** gdje\_smo\_u\_prvom\_nizu**))** **and**

489 gdje\_smo\_u\_drugom\_nizu **<** gornja\_granica **Then** **{**

490 **ValueAt(**pomocni\_niz **+** i**)**

491 **:** **=** **ValueAt(**originalni\_niz **+** gdje\_smo\_u\_drugom\_nizu**);**

492 gdje\_smo\_u\_drugom\_nizu **+=** **1;**

493 **}**

494 **Else** **{**

495 **ValueAt(**pomocni\_niz **+** i**)**

496 **:** **=** **ValueAt(**originalni\_niz **+** gdje\_smo\_u\_prvom\_nizu**);**

497 gdje\_smo\_u\_prvom\_nizu **+=** **1;**

498 **}**

499 **EndIf;**

500 i **+=** **1;**

501 **}**

502 **EndWhile;**

503 i**:**

504 **=** donja\_granica**;**

505 **While** i **<** gornja\_granica **Loop** **{**

506 **ValueAt(**originalni\_niz **+** i**)** **:** **=** **ValueAt(**pomocni\_niz **+** i**);**

507 i **+=** **1;**

508 **}**

509 **EndWhile;**

510 **Return;**

511 **}**

512 **Else** **{** **// QuickSort algoritam (nasumično ispremještani podnizovi)...**

513 broj\_pokretanja\_QuickSorta **+=** **1;**

514 **// Daljnji kod je približno prepisan s**

515 **// https://www.geeksforgeeks.org/quick-sort/**

516 **// Iskreno, ne razumijem ni ja točno kako funkcionira.**

517 **// On navodno preuređuje niz tako da svi elementi koji su manji**

518 **// od onog koji je bio prvi (pivot) dođu prije njega, a ostali**

519 **// poslije njega.**

520 **Integer32** pivot **:** **=** **ValueAt(**originalni\_niz **+** gornja\_granica **-** **1);**

521 **Integer32** i **:** **=** donja\_granica **-** **1;**

522 **Integer32** j **:** **=** donja\_granica**;**

523 **Integer32** pomocna\_varijabla\_za\_zamijenu**;**

524 **While** j **<** gornja\_granica **-** **1** **Loop** **{**

525 **If** **ValueAt(**originalni\_niz **+** j**)** **<** pivot **Then** **{**

526 i **+=** **1;**

527 pomocna\_varijabla\_za\_zamijenu**:**

528 **=** **ValueAt(**originalni\_niz **+** i**);**

529 **ValueAt(**originalni\_niz **+** i**)** **:** **=** **ValueAt(**originalni\_niz **+** j**);**

530 **ValueAt(**originalni\_niz **+** j**)** **:** **=** pomocna\_varijabla\_za\_zamijenu**;**

531 **}**

532 **EndIf;**

533 j **+=** **1;**

534 **}**

535 **EndWhile;**

536 pomocna\_varijabla\_za\_zamijenu**:**

537 **=** **ValueAt(**originalni\_niz **+** i **+** **1);**

538 **ValueAt(**originalni\_niz **+** i **+** **1)**

539 **:** **=** **ValueAt(**originalni\_niz **+** gornja\_granica **-** **1);**

540 **ValueAt(**originalni\_niz **+** gornja\_granica **-** **1)**

541 **:** **=** pomocna\_varijabla\_za\_zamijenu**;**

542 **Integer32** gdje\_je\_pivot **:** **=** i **+** **1;**

543 hybrid\_sort**(**donja\_granica**,** gdje\_je\_pivot**,** dubina\_rekurzije **+** **1);**

544 hybrid\_sort**(**gdje\_je\_pivot**,** gornja\_granica**,** dubina\_rekurzije **+** **1);**

545 **Return;**

546 **}**

547 **EndIf;**

548 **// Ovdje tok programa ne smije doći.**

549 printString**(**"Izgleda da compiler nije ispravno "

550 "preveo kontrolne strukture!"**);**

551 **}**

552 **EndFunction;**

553

554 **// Ovo je funkcija koju će pozvati JavaScript...**

555 **Function** pocetna\_AEC\_funkcija**()** **Which** **Returns** **Nothing** **Does** **{**

556 **If** originalni\_niz **=** **-1** **or** pomocni\_niz **=** **-1** **Then** **{**

557 **Return;** **// Ako JavaScript nastavlja pokretati ovaj program**

558 **// unatoč nedostatku memorije, neka onda on ne radi ništa.**

559 **}**

560 **EndIf;**

561 **// Testiraj matematičke funkcije...**

562 **If** abs**(**pow**(3,** **3)** **-** **27)** **>** **2** **Then** **{** **// Da, one su jako neprecizne, ali zato**

563 **// jako brze.**

564 printString**(**"Izgleda da matematicke funkcije ne funkcioniraju dobro."**);**

565 printString**(**"pow(3, 3) ="**);**

566 printFloat**(**pow**(3,** **3));**

567 **}**

568 **EndIf;**

569 **// Doznaj veličinu niza iz JavaScripta...**

570 **Integer32** prijasnja\_velicina\_niza **:** **=** velicina\_niza**;**

571 velicina\_niza**:**

572 **=** daj\_velicinu\_niza**();**

573 **// Ako je potrebno, zauzmi još memorije...**

574 **If** velicina\_niza **/** **(64** **\*** **1024** **/** **4)** **+**

575 **not(not(mod(**velicina\_niza**,** **64** **\*** **1024** **/** **4)))** **>**

576 prijasnja\_velicina\_niza **/** **(64** **\*** **1024** **/** **4)** **+**

577 **not(not(mod(**prijasnja\_velicina\_niza**,** **64** **\*** **1024** **/** **4)))** **or**

578 prijasnja\_velicina\_niza **=** **0** **Then** **{**

579 originalni\_niz**:**

580 **=** zauzmi\_memorijske\_stranice**(4** **\*** velicina\_niza **/** **(64** **\*** **1024)** **+**

581 **not(not(mod(**velicina\_niza**,** **64** **\*** **1024** **/** **4))));**

582 pomocni\_niz**:**

583 **=** zauzmi\_memorijske\_stranice**(4** **\*** velicina\_niza **/** **(64** **\*** **1024)** **+**

584 **not(not(mod(**velicina\_niza**,** **64** **\*** **1024** **/** **4))));**

585 **If** originalni\_niz **=** **-1** **or** pomocni\_niz **=** **-1** **Then** **{**

586 printString**(**"Nema dovoljno memorije za nastavak programa!?"**);**

587 izvijesti\_JavaScript\_o\_nedostatku\_memorije**();**

588 **Return;** **// Prekini izvršavanje ovog programa.**

589 **}**

590 **EndIf;**

591 **}**

592 **EndIf;**

593 **// Sada zatraži od JavaScripta da kopira niz koji treba poredati**

594 **// na memorijski prostor koji si (prethodno ili sada) zauzeo.**

595 kopiraj\_niz\_na\_adresu**(**originalni\_niz**);**

596 **// I sada ga kreni razvrstavati i mjeriti koliko ti treba vremena.**

597broj\_obrnuto\_poredanih\_podniza**:**

598 **=** broj\_vec\_poredanih\_podniza **:** **=** broj\_pokretanja\_QuickSorta

599 **:** **=** broj\_pokretanja\_MergeSorta **:** **=** broj\_pokretanja\_SelectSorta

600 **:** **=** **0;** **// Nisam mogao odoljeti da u svoj programski jezik ne dodam**

601 **// ulančano pridruživanje iz C-a, C++-a i JavaScripta (da možemo**

602 **// više varijabli postaviti na neku vrijednost u jednoj naredbi).**

603 broj\_mjerenja **+=** **1;**

604 pocni\_mjerenje\_vremena**();**

605 hybrid\_sort**(0,** velicina\_niza**,** **0);**

606 zavrsi\_mjerenje\_vremena**();**

607 **// Kad završi mjerenje vremena (koje se vrtilo u JavaScriptu),**

608 **// obavijesti JavaScript o onome što si ti izmjerio.**

609 izvijesti\_o\_obrnuto\_poredanim\_nizovima**(**broj\_obrnuto\_poredanih\_podniza**);**

610 izvijesti\_o\_poredanim\_nizovima**(**broj\_vec\_poredanih\_podniza**);**

611 izvijesti\_o\_pokretanju\_QuickSorta**(**broj\_pokretanja\_QuickSorta**);**

612 izvijesti\_o\_pokretanju\_MergeSorta**(**broj\_pokretanja\_MergeSorta**);**

613 izvijesti\_o\_pokretanju\_SelectSorta**(**broj\_pokretanja\_SelectSorta**);**

614 **// Napravi neki osnovni sanity-check, je li niz uistinu poredan?**

615 **Integer32** i **:** **=** **0;**

616 **While** i **<** velicina\_niza **-** **1** **Loop** **{**

617 **If** **ValueAt(**originalni\_niz **+** i**)** **>** **ValueAt(**originalni\_niz **+** i **+** **1)** **Then** **{**

618 printString**(**"Niz nije poredan!"**);**

619 **Return;** **// Nemoj to ispisati više puta, nego prekini program čim**

620 **// si uočio prvu nepodudarnost.**

621 **}**

622 **EndIf;**

623 i **+=** **1;**

624 **}**

625 **EndWhile;**

626 **}**

627 **EndFunction;**

628