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 **(**R"( ;; Sintaksu za višelinijske stringove preuzeo sam iz C++11-a, ta

62 ;; mi se sintaksa za višelinijske stringove nekako najviše sviđa.

63 (memory.grow

64 (i32.load %broj\_stranica) ;; Compiler će zamijeniti `%broj\_stranica` s

65 ;; asemblerskim kodom koji predstavlja pokazivač

66 ;; na varijablu `broj\_stranica`.

67 )

68 )"**);**

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

70 **// slobodne memorije...**

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

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

73 **}**

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

75 **// programski kôd.**

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

77 **// Mašini, jedna stranica**

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

79 **}**

80 **EndFunction;**

81

82 **Integer32** velicina\_niza**;**

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

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

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

86 **//"pomocni\_niz" su na početku**

87 **// programa nulti pokazivači.**

88

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

90 broj\_pokretanja\_QuickSorta**,** broj\_pokretanja\_MergeSorta**,**

91 broj\_pokretanja\_SelectSorta**;**

92

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

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

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

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

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

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

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

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

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

102 **// stack overflow.**

103

104 **Decimal32**

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

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

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

108

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

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

111

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

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

114 **EndIf;**

115 **If(**x **<** **8192** **and**

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

117 **// pojašnjenje ovoga...**

118 "(f32.nearest (f32.load %x))"**)]** **=** **0))** **Then** **{**

119 **Return**

120 memoizacija\_prirodnog\_logaritma**[asm\_f32(**"(f32.nearest (f32.load %x))"**)];**

121 **}**

122 **EndIf;**

123 **If** x **>** **8192** **Then** **{**

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

125 **}**

126 **EndIf;**

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

128 **// srednjoškolska matematika.**

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

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

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

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

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

134 **// čine nečitkima.**

135 **}**

136 **EndWhile;**

137 **If** x **<** **8192** **Then** **{**

138 **Return**

139 memoizacija\_prirodnog\_logaritma**[asm\_f32(**"(f32.nearest (f32.load %x))"**)]**

140 **:** **=** zbroj**;**

141 **}**

142 **EndIf;**

143 **Return** zbroj**;**

144 **}**

145 **EndFunction;**

146

147 **Decimal32**

148 memoizacija\_Eulerovog\_algoritma**[512];** **// Možda malo ubrza program, gotovo**

149 **// sigurno ga neće usporiti...**

150

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

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

153 **// što se definira.**

154

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

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

157 **not(**memoizacija\_Eulerovog\_algoritma

158 **[asm\_f32(**"(f32.nearest (f32.mul (f32.const 4) (f32.load %x)))"**)** **+**

159 **256]** **=** **0))**

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

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

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

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

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

165 **// compiler.**

166 **Return** memoizacija\_Eulerovog\_algoritma

167 **[asm\_f32(**"(f32.nearest (f32.mul (f32.const 4) (f32.load %x)))"**)** **+** **256];**

168 **}**

169 **EndIf;**

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

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

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

173 y **+=** epsilon **\*** y**;**

174 i **+=** epsilon**;**

175 **}**

176 **EndWhile;**

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

178 **Return** memoizacija\_Eulerovog\_algoritma

179 **[asm\_f32(**"(f32.nearest (f32.mul (f32.const 4) (f32.load %x)))"**)** **+** **256]**

180 **:** **=** y**;**

181 **}**

182 **EndIf;**

183 **Return** y**;**

184 **}**

185 **EndFunction;**

186

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

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

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

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

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

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

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

194 **// na desno nego na lijevo.**

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

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

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

198 **}**

199 **EndFunction;**

200

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

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

203 **If** DEBUG **=** **1** **Then** **{**

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

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

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

207 printFloat**(**x**);**

208 printFloat**(**y**);**

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

210 **}**

211 **EndIf;**

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

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

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

215 **}**

216 **EndIf;**

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

218 **// riskiram da će JavaScript**

219 **// virtualna mašina prekinuti**

220 **// izvođenje programa jer je**

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

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

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

224 **// milijarde u pozitivno i**

225 **// negativno).**

226 **}**

227 **EndFunction;**

228

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

230 **Decimal32** result

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

232 **Return** x **=**

233 **0** **?** **0** **:** ostatak\_pri\_dijeljenju**(**x**,** **2)** **=** **1** **and** x **<** **0** **?** **-**result **:** result**;**

234 **}**

235 **EndFunction;**

236

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

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

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

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

241 **// 2 (0 ili 1), znači da je već**

242 **// poredan, pa prekidamo**

243 **// izvođenje ovog potprograma.**

244 **Return;**

245 **}**

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

247 **// vrijedi ga posebno**

248 **// obraditi jer time**

249 **// možemo znatno ubrzati**

250 **// program.**

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

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

253 **Integer32** pomocna\_varijabla\_za\_zamijenu

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

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

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

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

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

259 **}**

260 **EndIf;**

261 **Return;**

262 **}**

263 **ElseIf** gornja\_granica **-**

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

265 **73** **Then** **{**

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

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

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

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

270 broj\_pokretanja\_SelectSorta**:**

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

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

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

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

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

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

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

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

279 gdje\_je\_minimum

280 **/\***

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

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

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

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

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

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

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

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

289 **\*/**

290 **:**

291 **=** j**;**

292 **}**

293 **EndIf;**

294 j **+=** **1;**

295 **}**

296 **EndWhile;**

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

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

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

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

301 i **+=** **1;**

302 **}**

303 **EndWhile;**

304 **Return;**

305 **}**

306 **EndIf;**

307 **Decimal32** razvrstanost **:** **=** **0;**

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

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

310 razvrstanost **+=**

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

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

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

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

315 **=** **0;**

316 **}**

317 **EndIf;**

318 i **+=** **1;**

319 **}**

320 **EndWhile;**

321 razvrstanost**:**

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

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

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

325 **// To ne smije biti...**

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

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

328 **// debuggerima za JavaScript**

329 **// virtualnu mašinu ako dođe**

330 **// do problema, lakše mi**

331 **// je ispisati brojeve u**

332 **// programu nego tražiti**

333 **// kako narediti**

334 **// debuggeru da ih**

335 **// ispiše.**

336 i**:**

337 **=** donja\_granica**;**

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

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

340 i **+=** **1;**

341 **}**

342 **EndWhile;**

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

344 **}**

345 **EndIf;**

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

347 **// Opet ne smije biti...**

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

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

350 i**:**

351 **=** donja\_granica**;**

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

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

354 i **+=** **1;**

355 **}**

356 **EndWhile;**

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

358 **}**

359 **EndIf;**

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

361 **Then** **{**

362 **// Open ne smije biti...**

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

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

365 i**:**

366 **=** donja\_granica**;**

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

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

369 i **+=** **1;**

370 **}**

371 **EndWhile;**

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

373 **}**

374 **EndIf;**

375 **// Idemo dalje...**

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

377 **// izračunati ako ne**

378 **// pozivamo "pow" gdje**

379 **// ne treba (kad je**

380 **// eksponent prirodan**

381 **// broj).**

382i**:**

383 **=** **1;**

384 **While** i **<** **8** **Loop** **{**

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

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

387 i **+=** **1;**

388 **}**

389 **EndWhile;**

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

391 **// usporedbi QuickSort napraviti:**

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

393 **Decimal32** polinom\_pod\_apsolutnom

394 **:** **=** **2.38854** **\*** razvrstanost\_na\_potenciju**[7]** **-**

395 **0.284258** **\*** razvrstanost\_na\_potenciju**[6]** **-**

396 **1.87104** **\*** razvrstanost\_na\_potenciju**[5]** **+**

397 **0.372637** **\*** razvrstanost\_na\_potenciju**[4]** **+**

398 **0.167242** **\*** razvrstanost\_na\_potenciju**[3]** **-**

399 **0.0884977** **\*** razvrstanost\_na\_potenciju**[2]** **+** **0.315119** **\*** razvrstanost**;**

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

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

402 prirodni\_logaritam**(**

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

404 **1.05** **+**

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

406 prirodni\_logaritam**(**

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

408 prirodni\_logaritam**(2))** **\***

409 **0.9163** **\*** abs**(**polinom\_pod\_apsolutnom**);**

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

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

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

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

414 **(**gornja\_granica **-**

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

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

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

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

419 **// dijelova originalnog niza u**

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

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

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

423 **// da je cjelokupni algoritam brži**

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

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

426 **// pa ćemo preciznijim mjerenjima**

427 **// vidjeti što je bolje.**

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

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

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

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

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

433 **Return;**

434 **}**

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

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

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

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

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

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

441 j **-=** **1;**

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

443 **}**

444 **EndWhile;**

445 i**:**

446 **=** donja\_granica**;**

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

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

449 i **+=** **1;**

450 **}**

451 **EndWhile;**

452 **Return;**

453 **}**

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

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

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

457 prirodni\_logaritam**(2))**

458 **// JavaScriptina virtualna mašina ima**

459 **// 4KB memorije na sistemskom stogu,**

460 **// i alociranje više heap memorije**

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

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

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

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

465 **)** **and**

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

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

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

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

470 **Then** **{**

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

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

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

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

475 **// nije opcija)...**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

493 sredina\_niza **or**

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

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

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

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

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

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

500 **}**

501 **Else** **{**

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

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

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

505 **}**

506 **EndIf;**

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

508 **}**

509 **EndWhile;**

510 i**:**

511 **=** donja\_granica**;**

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

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

514 i **+=** **1;**

515 **}**

516 **EndWhile;**

517 **Return;**

518 **}**

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

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

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

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

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

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

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

526 **// poslije njega.**

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

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

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

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

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

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

533 i **+=** **1;**

534 pomocna\_varijabla\_za\_zamijenu**:**

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

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

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

538 **}**

539 **EndIf;**

540 j **+=** **1;**

541 **}**

542 **EndWhile;**

543 pomocna\_varijabla\_za\_zamijenu**:**

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

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

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

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

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

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

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

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

552 **Return;**

553 **}**

554 **EndIf;**

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

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

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

558 **}**

559 **EndFunction;**

560

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

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

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

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

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

566 **}**

567 **EndIf;**

568 **// Testiraj matematičke funkcije...**

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

570 **// jako brze.**

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

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

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

574 **}**

575 **EndIf;**

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

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

578 velicina\_niza**:**

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

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

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

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

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

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

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

586 originalni\_niz**:**

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

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

589 pomocni\_niz**:**

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

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

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

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

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

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

596 **}**

597 **EndIf;**

598 **}**

599 **EndIf;**

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

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

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

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

604broj\_obrnuto\_poredanih\_podniza**:**

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

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

607 **:** **=** **0;** **// Nisam mogao odoljeti da u svoj programski jezik ne dodam**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

622 **Integer32** i **:** **=** **0;**

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

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

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

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

627 **// si uočio prvu nepodudarnost.**

628 **}**

629 **EndIf;**

630 i **+=** **1;**

631 **}**

632 **EndWhile;**

633 **}**

634 **EndFunction;**

635