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**[512];** **// 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 (f32.mul (f32.const 4) (local.get 0)))"**)** **+**

155 **256]** **=** **0))**

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

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

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

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

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

161 **// compiler.**

162 **Return** memoizacija\_Eulerovog\_algoritma

163 **[asm\_f32(**"(f32.nearest (f32.mul (f32.const 4) (local.get 0)))"**)** **+** **256];**

164 **}**

165 **EndIf;**

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

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

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

169 y **+=** epsilon **\*** y**;**

170 i **+=** epsilon**;**

171 **}**

172 **EndWhile;**

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

174 **Return** memoizacija\_Eulerovog\_algoritma

175 **[asm\_f32(**"(f32.nearest (f32.mul (f32.const 4) (local.get 0)))"**)** **+** **256]**

176 **:** **=** y**;**

177 **}**

178 **EndIf;**

179 **Return** y**;**

180 **}**

181 **EndFunction;**

182

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

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

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

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

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

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

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

190 **// na desno nego na lijevo.**

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

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

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

194 **}**

195 **EndFunction;**

196

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

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

199 **If** DEBUG **=** **1** **Then** **{**

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

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

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

203 printFloat**(**x**);**

204 printFloat**(**y**);**

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

206 **}**

207 **EndIf;**

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

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

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

211 **}**

212 **EndIf;**

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

214 **// riskiram da će JavaScript**

215 **// virtualna mašina prekinuti**

216 **// izvođenje programa jer je**

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

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

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

220 **// milijarde u pozitivno i**

221 **// negativno).**

222 **}**

223 **EndFunction;**

224

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

226 **Decimal32** result

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

228 **Return** x **=**

229 **0** **?** **0** **:** ostatak\_pri\_dijeljenju**(**x**,** **2)** **=** **1** **and** x **<** **0** **?** **-**result **:** result**;**

230 **}**

231 **EndFunction;**

232

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

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

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

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

237 **// 2 (0 ili 1), znači da je već**

238 **// poredan, pa prekidamo**

239 **// izvođenje ovog potprograma.**

240 **Return;**

241 **}**

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

243 **// vrijedi ga posebno**

244 **// obraditi jer time**

245 **// možemo znatno ubrzati**

246 **// program.**

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

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

249 **Integer32** pomocna\_varijabla\_za\_zamijenu

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

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

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

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

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

255 **}**

256 **EndIf;**

257 **Return;**

258 **}**

259 **ElseIf** gornja\_granica **-**

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

261 **73** **Then** **{**

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

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

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

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

266 broj\_pokretanja\_SelectSorta**:**

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

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

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

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

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

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

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

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

275 gdje\_je\_minimum

276 **/\***

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

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

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

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

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

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

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

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

285 **\*/**

286 **:**

287 **=** j**;**

288 **}**

289 **EndIf;**

290 j **+=** **1;**

291 **}**

292 **EndWhile;**

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

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

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

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

297 i **+=** **1;**

298 **}**

299 **EndWhile;**

300 **Return;**

301 **}**

302 **EndIf;**

303 **Decimal32** razvrstanost **:** **=** **0;**

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

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

306 razvrstanost **+=**

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

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

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

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

311 **=** **0;**

312 **}**

313 **EndIf;**

314 i **+=** **1;**

315 **}**

316 **EndWhile;**

317 razvrstanost**:**

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

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

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

321 **// To ne smije biti...**

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

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

324 **// debuggerima za JavaScript**

325 **// virtualnu mašinu ako dođe**

326 **// do problema, lakše mi**

327 **// je ispisati brojeve u**

328 **// programu nego tražiti**

329 **// kako narediti**

330 **// debuggeru da ih**

331 **// ispiše.**

332 i**:**

333 **=** donja\_granica**;**

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

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

336 i **+=** **1;**

337 **}**

338 **EndWhile;**

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

340 **}**

341 **EndIf;**

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

343 **// Opet ne smije biti...**

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

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

346 i**:**

347 **=** donja\_granica**;**

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

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

350 i **+=** **1;**

351 **}**

352 **EndWhile;**

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

354 **}**

355 **EndIf;**

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

357 **Then** **{**

358 **// Open ne smije biti...**

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

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

361 i**:**

362 **=** donja\_granica**;**

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

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

365 i **+=** **1;**

366 **}**

367 **EndWhile;**

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

369 **}**

370 **EndIf;**

371 **// Idemo dalje...**

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

373 **// izračunati ako ne**

374 **// pozivamo "pow" gdje**

375 **// ne treba (kad je**

376 **// eksponent prirodan**

377 **// broj).**

378i**:**

379 **=** **1;**

380 **While** i **<** **8** **Loop** **{**

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

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

383 i **+=** **1;**

384 **}**

385 **EndWhile;**

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

387 **// usporedbi QuickSort napraviti:**

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

389 **Decimal32** polinom\_pod\_apsolutnom

390 **:** **=** **2.38854** **\*** razvrstanost\_na\_potenciju**[7]** **-**

391 **0.284258** **\*** razvrstanost\_na\_potenciju**[6]** **-**

392 **1.87104** **\*** razvrstanost\_na\_potenciju**[5]** **+**

393 **0.372637** **\*** razvrstanost\_na\_potenciju**[4]** **+**

394 **0.167242** **\*** razvrstanost\_na\_potenciju**[3]** **-**

395 **0.0884977** **\*** razvrstanost\_na\_potenciju**[2]** **+** **0.315119** **\*** razvrstanost**;**

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

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

398 prirodni\_logaritam**(**

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

400 **1.05** **+**

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

402 prirodni\_logaritam**(**

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

404 prirodni\_logaritam**(2))** **\***

405 **0.9163** **\*** abs**(**polinom\_pod\_apsolutnom**);**

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

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

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

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

410 **(**gornja\_granica **-**

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

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

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

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

415 **// dijelova originalnog niza u**

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

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

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

419 **// da je cjelokupni algoritam brži**

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

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

422 **// pa ćemo preciznijim mjerenjima**

423 **// vidjeti što je bolje.**

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

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

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

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

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

429 **Return;**

430 **}**

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

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

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

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

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

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

437 j **-=** **1;**

438 i **+=** **1;**

439 **}**

440 **EndWhile;**

441 i**:**

442 **=** donja\_granica**;**

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

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

445 i **+=** **1;**

446 **}**

447 **EndWhile;**

448 **Return;**

449 **}**

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

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

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

453 prirodni\_logaritam**(2))**

454 **// JavaScriptina virtualna mašina ima**

455 **// 4KB memorije na sistemskom stogu,**

456 **// i alociranje više heap memorije**

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

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

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

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

461 **)** **and**

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

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

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

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

466 **Then** **{**

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

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

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

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

471 **// nije opcija)...**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

489 sredina\_niza **or**

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

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

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

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

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

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

496 **}**

497 **Else** **{**

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

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

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

501 **}**

502 **EndIf;**

503 i **+=** **1;**

504 **}**

505 **EndWhile;**

506 i**:**

507 **=** donja\_granica**;**

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

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

510 i **+=** **1;**

511 **}**

512 **EndWhile;**

513 **Return;**

514 **}**

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

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

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

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

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

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

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

522 **// poslije njega.**

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

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

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

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

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

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

529 i **+=** **1;**

530 pomocna\_varijabla\_za\_zamijenu**:**

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

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

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

534 **}**

535 **EndIf;**

536 j **+=** **1;**

537 **}**

538 **EndWhile;**

539 pomocna\_varijabla\_za\_zamijenu**:**

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

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

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

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

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

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

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

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

548 **Return;**

549 **}**

550 **EndIf;**

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

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

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

554 **}**

555 **EndFunction;**

556

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

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

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

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

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

562 **}**

563 **EndIf;**

564 **// Testiraj matematičke funkcije...**

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

566 **// jako brze.**

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

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

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

570 **}**

571 **EndIf;**

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

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

574 velicina\_niza**:**

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

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

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

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

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

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

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

582 originalni\_niz**:**

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

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

585 pomocni\_niz**:**

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

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

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

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

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

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

592 **}**

593 **EndIf;**

594 **}**

595 **EndIf;**

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

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

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

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

600broj\_obrnuto\_poredanih\_podniza**:**

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

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

603 **:** **=** **0;** **// Nisam mogao odoljeti da u svoj programski jezik ne dodam**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

618 **Integer32** i **:** **=** **0;**

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

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

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

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

623 **// si uočio prvu nepodudarnost.**

624 **}**

625 **EndIf;**

626 i **+=** **1;**

627 **}**

628 **EndWhile;**

629 **}**

630 **EndFunction;**

631