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;**

68 **}**

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

70 **// programski kôd.**

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

72 **// Mašini, jedna stranica**

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

74 **}**

75 **EndFunction;**

76

77 **Integer32** velicina\_niza**;**

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

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

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

81 **//"pomocni\_niz" su na početku**

82 **// programa nulti pokazivači.**

83

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

85 broj\_pokretanja\_QuickSorta**,** broj\_pokretanja\_MergeSorta**,**

86 broj\_pokretanja\_SelectSorta**;**

87

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

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

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

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

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

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

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

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

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

97 **// stack overflow.**

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

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

100 **// srednjoškolska matematika.**

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

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

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

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

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

106 **// čine nečitkima.**

107 **}**

108 **EndWhile;**

109 **Return** zbroj**;**

110 **}**

111 **EndFunction;**

112

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

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

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

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

117 y **+=** epsilon **\*** y**;**

118 i **+=** epsilon**;**

119 **}**

120 **EndWhile;**

121 **Return** y**;**

122 **}**

123 **EndFunction;**

124

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

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

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

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

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

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

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

132 **// na desno nego na lijevo.**

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

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

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

136 **}**

137 **EndFunction;**

138

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

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

141 **If** DEBUG **=** **1** **Then** **{**

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

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

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

145 printFloat**(**x**);**

146 printFloat**(**y**);**

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

148 **}**

149 **EndIf;**

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

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

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

153 **}**

154 **EndIf;**

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

156 **// riskiram da će JavaScript**

157 **// virtualna mašina prekinuti**

158 **// izvođenje programa jer je**

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

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

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

162 **// milijarde u pozitivno i**

163 **// negativno).**

164 **}**

165 **EndFunction;**

166

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

168 **Decimal32** result

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

170 **Return** x **=**

171 **0** **?** **0** **:** ostatak\_pri\_dijeljenju**(**x**,** **2)** **=** **1** **and** x **<** **0** **?** **-**result **:** result**;**

172 **}**

173 **EndFunction;**

174

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

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

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

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

179 **// 2 (0 ili 1), znači da je već**

180 **// poredan, pa prekidamo**

181 **// izvođenje ovog potprograma.**

182 **Return;**

183 **}**

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

185 **// vrijedi ga posebno**

186 **// obraditi jer time**

187 **// možemo znatno ubrzati**

188 **// program.**

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

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

191 **Integer32** pomocna\_varijabla\_za\_zamijenu

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

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

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

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

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

197 **}**

198 **EndIf;**

199 **Return;**

200 **}**

201 **ElseIf** gornja\_granica **-**

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

203 **73** **Then** **{**

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

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

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

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

208 broj\_pokretanja\_SelectSorta**:**

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

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

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

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

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

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

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

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

217 gdje\_je\_minimum

218 **/\***

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

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

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

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

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

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

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

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

227 **\*/**

228 **:**

229 **=** j**;**

230 **}**

231 **EndIf;**

232 j **+=** **1;**

233 **}**

234 **EndWhile;**

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

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

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

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

239 i **+=** **1;**

240 **}**

241 **EndWhile;**

242 **Return;**

243 **}**

244 **EndIf;**

245 **Decimal32** razvrstanost **:** **=** **0;**

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

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

248 razvrstanost **+=**

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

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

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

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

253 **=** **0;**

254 **}**

255 **EndIf;**

256 i **+=** **1;**

257 **}**

258 **EndWhile;**

259 razvrstanost**:**

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

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

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

263 **// To ne smije biti...**

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

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

266 **// debuggerima za JavaScript**

267 **// virtualnu mašinu ako dođe**

268 **// do problema, lakše mi**

269 **// je ispisati brojeve u**

270 **// programu nego tražiti**

271 **// kako narediti**

272 **// debuggeru da ih**

273 **// ispiše.**

274 i**:**

275 **=** donja\_granica**;**

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

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

278 i **+=** **1;**

279 **}**

280 **EndWhile;**

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

282 **}**

283 **EndIf;**

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

285 **// Opet ne smije biti...**

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

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

288 i**:**

289 **=** donja\_granica**;**

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

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

292 i **+=** **1;**

293 **}**

294 **EndWhile;**

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

296 **}**

297 **EndIf;**

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

299 **Then** **{**

300 **// Open ne smije biti...**

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

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

303 i**:**

304 **=** donja\_granica**;**

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

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

307 i **+=** **1;**

308 **}**

309 **EndWhile;**

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

311 **}**

312 **EndIf;**

313 **// Idemo dalje...**

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

315 **// izračunati ako ne**

316 **// pozivamo "pow" gdje**

317 **// ne treba (kad je**

318 **// eksponent prirodan**

319 **// broj).**

320i**:**

321 **=** **1;**

322 **While** i **<** **8** **Loop** **{**

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

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

325 i **+=** **1;**

326 **}**

327 **EndWhile;**

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

329 **// usporedbi QuickSort napraviti:**

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

331 **Decimal32** polinom\_pod\_apsolutnom

332 **:** **=** **2.38854** **\*** razvrstanost\_na\_potenciju**[7]** **-**

333 **0.284258** **\*** razvrstanost\_na\_potenciju**[6]** **-**

334 **1.87104** **\*** razvrstanost\_na\_potenciju**[5]** **+**

335 **0.372637** **\*** razvrstanost\_na\_potenciju**[4]** **+**

336 **0.167242** **\*** razvrstanost\_na\_potenciju**[3]** **-**

337 **0.0884977** **\*** razvrstanost\_na\_potenciju**[2]** **+** **0.315119** **\*** razvrstanost**;**

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

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

340 prirodni\_logaritam**(**

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

342 **1.05** **+**

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

344 prirodni\_logaritam**(**

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

346 prirodni\_logaritam**(2))** **\***

347 **0.9163** **\*** abs**(**polinom\_pod\_apsolutnom**);**

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

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

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

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

352 **(**gornja\_granica **-**

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

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

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

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

357 **// dijelova originalnog niza u**

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

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

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

361 **// da je cjelokupni algoritam brži**

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

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

364 **// pa ćemo preciznijim mjerenjima**

365 **// vidjeti što je bolje.**

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

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

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

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

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

371 **Return;**

372 **}**

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

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

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

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

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

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

379 j **-=** **1;**

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

381 **}**

382 **EndWhile;**

383 i**:**

384 **=** donja\_granica**;**

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

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

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

388 **}**

389 **EndWhile;**

390 **Return;**

391 **}**

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

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

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

395 prirodni\_logaritam**(2))**

396 **// JavaScriptina virtualna mašina ima**

397 **// 4KB memorije na sistemskom stogu,**

398 **// i alociranje više heap memorije**

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

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

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

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

403 **)** **and**

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

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

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

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

408 **Then** **{**

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

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

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

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

413 **// nije opcija)...**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

431 sredina\_niza **or**

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

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

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

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

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

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

438 **}**

439 **Else** **{**

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

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

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

443 **}**

444 **EndIf;**

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

446 **}**

447 **EndWhile;**

448 i**:**

449 **=** donja\_granica**;**

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

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

452 i **+=** **1;**

453 **}**

454 **EndWhile;**

455 **Return;**

456 **}**

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

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

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

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

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

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

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

464 **// poslije njega.**

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

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

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

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

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

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

471 i **+=** **1;**

472 pomocna\_varijabla\_za\_zamijenu**:**

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

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

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

476 **}**

477 **EndIf;**

478 j **+=** **1;**

479 **}**

480 **EndWhile;**

481 pomocna\_varijabla\_za\_zamijenu**:**

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

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

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

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

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

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

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

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

490 **Return;**

491 **}**

492 **EndIf;**

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

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

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

496 **}**

497 **EndFunction;**

498

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

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

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

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

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

504 **}**

505 **EndIf;**

506 **// Testiraj matematičke funkcije...**

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

508 **// jako brze.**

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

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

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

512 **}**

513 **EndIf;**

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

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

516 velicina\_niza**:**

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

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

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

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

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

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

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

524 originalni\_niz**:**

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

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

527 pomocni\_niz**:**

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

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

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

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

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

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

534 **}**

535 **EndIf;**

536 **}**

537 **EndIf;**

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

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

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

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

542broj\_obrnuto\_poredanih\_podniza**:**

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

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

545 **:** **=** **0;** **// Nisam mogao odoljeti da u svoj programski jezik ne dodam**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

560 **Integer32** i **:** **=** **0;**

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

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

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

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

565 **// si uočio prvu nepodudarnost.**

566 **}**

567 **EndIf;**

568 i **+=** **1;**

569 **}**

570 **EndWhile;**

571 **}**

572 **EndFunction;**

573