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 **PointerToInteger32** adresa**)** **Which** **Returns** **Nothing** **Is** **External;**

20 **Function** printString**(PointerToCharacter** 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 **PointerToCharacter**

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"multiline(

62 ;; Sintaksu za višelinijske stringove preuzeo sam iz C++11-a, ta mi

63 ;; se sintaksa za višelinijske stringove nekako najviše sviđa.

64 (memory.grow

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

66 ;; asemblerskim kodom koji predstavlja pokazivač

67 ;; na varijablu `broj\_stranica`. Razumijem da

68 ;; način na koji GCC i CLANG compileri za C++

69 ;; implementiraju inline assembly ima svoje

70 ;; prednosti, ali to je teže i za koristiti

71 ;; (pogotovo za ovakve jednostavne umetnute

72 ;; asemblerske kodove) i za implementirati u

73 ;; compileru.

74 )

75 )multiline"**);**

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

77 **// slobodne memorije...**

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

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

80 **}**

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

82 **// programski kôd.**

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

84 **// Mašini, jedna stranica**

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

86 **}**

87 **EndFunction;**

88

89 **Integer32** velicina\_niza**;**

90 **PointerToInteger32** originalni\_niz**,**

91 pomocni\_niz**;** **// To su globalne varijable, po defaultu su u nuli, dakle**

92 **// "originalni\_niz" i "pomocni\_niz" su na početku programa**

93 **// nulti pokazivači.**

94

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

96 broj\_pokretanja\_QuickSorta**,** broj\_pokretanja\_MergeSorta**,**

97 broj\_pokretanja\_SelectSorta**;**

98

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

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

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

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

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

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

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

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

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

108 **// stack overflow.**

109

110 **Decimal32**

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

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

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

114

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

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

117

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

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

120 **EndIf;**

121 **If(**x **<** **8192** **and**

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

123 **// pojašnjenje ovoga...**

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

125 **Return**

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

127 **}**

128 **EndIf;**

129 **If** x **>** **8192** **Then** **{**

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

131 **}**

132 **EndIf;**

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

134 **// srednjoškolska matematika.**

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

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

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

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

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

140 **// čine nečitkima.**

141 **}**

142 **EndWhile;**

143 **If** x **<** **8192** **Then** **{**

144 **Return**

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

146 **:** **=** zbroj**;**

147 **}**

148 **EndIf;**

149 **Return** zbroj**;**

150 **}**

151 **EndFunction;**

152

153 **Decimal32**

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

155 **// sigurno ga neće usporiti...**

156

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

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

159 **// što se definira.**

160

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

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

163 **not(**memoizacija\_Eulerovog\_algoritma

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

165 **256]** **=** **0))**

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

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

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

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

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

171 **// compiler.**

172 **Return** memoizacija\_Eulerovog\_algoritma

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

174 **}**

175 **EndIf;**

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

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

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

179 y **+=** epsilon **\*** y**;**

180 i **+=** epsilon**;**

181 **}**

182 **EndWhile;**

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

184 **Return** memoizacija\_Eulerovog\_algoritma

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

186 **:** **=** y**;**

187 **}**

188 **EndIf;**

189 **Return** y**;**

190 **}**

191 **EndFunction;**

192

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

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

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

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

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

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

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

200 **// na desno nego na lijevo.**

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

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

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

204 **}**

205 **EndFunction;**

206

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

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

209 **If** DEBUG **=** **1** **Then** **{**

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

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

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

213 printFloat**(**x**);**

214 printFloat**(**y**);**

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

216 **}**

217 **EndIf;**

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

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

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

221 **}**

222 **EndIf;**

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

224 **// riskiram da će JavaScript**

225 **// virtualna mašina prekinuti**

226 **// izvođenje programa jer je**

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

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

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

230 **// milijarde u pozitivno i**

231 **// negativno).**

232 **}**

233 **EndFunction;**

234

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

236 **Decimal32** result

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

238 **Return** x **=**

239 **0** **?** **0** **:** ostatak\_pri\_dijeljenju**(**x**,** **2)** **=** **1** **and** x **<** **0** **?** **-**result **:** result**;**

240 **}**

241 **EndFunction;**

242

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

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

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

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

247 **// 2 (0 ili 1), znači da je već**

248 **// poredan, pa prekidamo**

249 **// izvođenje ovog potprograma.**

250 **Return;**

251 **}**

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

253 **// vrijedi ga posebno**

254 **// obraditi jer time**

255 **// možemo znatno ubrzati**

256 **// program.**

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

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

259 **Integer32** pomocna\_varijabla\_za\_zamijenu

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

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

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

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

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

265 **}**

266 **EndIf;**

267 **Return;**

268 **}**

269 **ElseIf** gornja\_granica **-**

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

271 **73** **Then** **{**

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

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

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

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

276 broj\_pokretanja\_SelectSorta**:**

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

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

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

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

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

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

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

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

285 gdje\_je\_minimum

286 **/\***

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

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

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

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

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

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

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

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

295 **\*/**

296 **:**

297 **=** j**;**

298 **}**

299 **EndIf;**

300 j **+=** **1;**

301 **}**

302 **EndWhile;**

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

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

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

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

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

308 **}**

309 **EndWhile;**

310 **Return;**

311 **}**

312 **EndIf;**

313 **Decimal32** razvrstanost **:** **=** **0;**

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

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

316 razvrstanost **+=**

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

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

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

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

321 **=** **0;**

322 **}**

323 **EndIf;**

324 i **+=** **1;**

325 **}**

326 **EndWhile;**

327 razvrstanost**:**

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

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

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

331 **// To ne smije biti...**

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

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

334 **// debuggerima za JavaScript**

335 **// virtualnu mašinu ako dođe**

336 **// do problema, lakše mi**

337 **// je ispisati brojeve u**

338 **// programu nego tražiti**

339 **// kako narediti**

340 **// debuggeru da ih**

341 **// ispiše.**

342 i**:**

343 **=** donja\_granica**;**

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

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

346 i **+=** **1;**

347 **}**

348 **EndWhile;**

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

350 **}**

351 **EndIf;**

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

353 **// Opet ne smije biti...**

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

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

356 i**:**

357 **=** donja\_granica**;**

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

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

360 i **+=** **1;**

361 **}**

362 **EndWhile;**

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

364 **}**

365 **EndIf;**

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

367 **Then** **{**

368 **// Open ne smije biti...**

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

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

371 i**:**

372 **=** donja\_granica**;**

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

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

375 i **+=** **1;**

376 **}**

377 **EndWhile;**

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

379 **}**

380 **EndIf;**

381 **// Idemo dalje...**

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

383 **// izračunati ako ne**

384 **// pozivamo "pow" gdje**

385 **// ne treba (kad je**

386 **// eksponent prirodan**

387 **// broj).**

388i**:**

389 **=** **1;**

390 **While** i **<** **8** **Loop** **{**

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

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

393 i **+=** **1;**

394 **}**

395 **EndWhile;**

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

397 **// usporedbi QuickSort napraviti:**

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

399 **Decimal32** polinom\_pod\_apsolutnom

400 **:** **=** **2.38854** **\*** razvrstanost\_na\_potenciju**[7]** **-**

401 **0.284258** **\*** razvrstanost\_na\_potenciju**[6]** **-**

402 **1.87104** **\*** razvrstanost\_na\_potenciju**[5]** **+**

403 **0.372637** **\*** razvrstanost\_na\_potenciju**[4]** **+**

404 **0.167242** **\*** razvrstanost\_na\_potenciju**[3]** **-**

405 **0.0884977** **\*** razvrstanost\_na\_potenciju**[2]** **+** **0.315119** **\*** razvrstanost**;**

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

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

408 prirodni\_logaritam**(**

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

410 **1.05** **+**

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

412 prirodni\_logaritam**(**

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

414 prirodni\_logaritam**(2))** **\***

415 **0.9163** **\*** abs**(**polinom\_pod\_apsolutnom**);**

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

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

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

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

420 **(**gornja\_granica **-**

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

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

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

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

425 **// dijelova originalnog niza u**

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

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

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

429 **// da je cjelokupni algoritam brži**

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

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

432 **// pa ćemo preciznijim mjerenjima**

433 **// vidjeti što je bolje.**

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

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

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

437 **If** razvrstanost **>=** **1** **Then** **{**

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

439 **Return;**

440 **}**

441 **ElseIf** razvrstanost **<=** **-1** **Then** **{**

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

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

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

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

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

447 j **-=** **1;**

448 i **+=** **1;**

449 **}**

450 **EndWhile;**

451 i**:**

452 **=** donja\_granica**;**

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

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

455 i **+=** **1;**

456 **}**

457 **EndWhile;**

458 **Return;**

459 **}**

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

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

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

463 prirodni\_logaritam**(2))**

464 **// JavaScriptina virtualna mašina ima**

465 **// 4KB memorije na sistemskom stogu,**

466 **// i alociranje više heap memorije**

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

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

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

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

471 **)** **and**

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

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

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

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

476 **Then** **{**

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

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

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

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

481 **// nije opcija)...**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

499 sredina\_niza **or**

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

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

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

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

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

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

506 **}**

507 **Else** **{**

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

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

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

511 **}**

512 **EndIf;**

513 i **+=** **1;**

514 **}**

515 **EndWhile;**

516 i**:**

517 **=** donja\_granica**;**

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

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

520 i **+=** **1;**

521 **}**

522 **EndWhile;**

523 **Return;**

524 **}**

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

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

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

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

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

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

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

532 **// poslije njega.**

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

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

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

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

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

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

539 i **+=** **1;**

540 pomocna\_varijabla\_za\_zamijenu**:**

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

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

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

544 **}**

545 **EndIf;**

546 j **+=** **1;**

547 **}**

548 **EndWhile;**

549 pomocna\_varijabla\_za\_zamijenu**:**

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

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

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

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

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

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

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

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

558 **Return;**

559 **}**

560 **EndIf;**

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

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

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

564 **}**

565 **EndFunction;**

566

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

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

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

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

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

572 **}**

573 **EndIf;**

574 **// Testiraj matematičke funkcije...**

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

576 **// jako brze.**

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

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

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

580 **}**

581 **EndIf;**

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

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

584 velicina\_niza**:**

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

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

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

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

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

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

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

592 originalni\_niz**:**

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

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

595 pomocni\_niz**:**

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

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

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

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

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

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

602 **}**

603 **EndIf;**

604 **}**

605 **EndIf;**

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

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

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

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

610broj\_obrnuto\_poredanih\_podniza**:**

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

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

613 **:** **=** **0;** **// Nisam mogao odoljeti da u svoj programski jezik ne dodam**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

628 **Integer32** i **:** **=** **0;**

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

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

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

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

633 **// si uočio prvu nepodudarnost.**

634 **}**

635 **EndIf;**

636 i **+=** **1;**

637 **}**

638 **EndWhile;**

639 **}**

640 **EndFunction;**

641