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\_o\_broju\_usporedbi**(Integer32** n**)** **Which** **Returns** **Nothing** **Is** **External;**

37 **Function**

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

39

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

41 **// pokretanje raznih algoritama**

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

43 **//što dobro dođe za**

44 **// eksperimentiranje s algoritmima.**

45

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

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

48 **PointerToCharacter**

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

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

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

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

53 **Integer32** nova\_adresa\_u\_stranicama

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

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

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

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

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

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

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

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

62 **// 64-bitni način rada nije baš zanemariva.**

63 **(**R"multiline(

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

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

66 (memory.grow

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

68 ;; asemblerskim kodom koji predstavlja pokazivač

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

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

71 ;; implementiraju inline assembly ima svoje

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

73 ;; (pogotovo za ovakve jednostavne umetnute

74 ;; asemblerske kodove) i za implementirati u

75 ;; compileru.

76 )

77 )multiline"**);**

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

79 **// slobodne memorije...**

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

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

82 **}**

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

84 **// programski kôd.**

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

86 **// Mašini, jedna stranica**

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

88 **}**

89 **EndFunction;**

90

91 **Integer32** velicina\_niza**;**

92 **PointerToInteger32** originalni\_niz**,**

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

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

95 **// nulti pokazivači.**

96

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

98 broj\_pokretanja\_QuickSorta**,** broj\_pokretanja\_MergeSorta**,**

99 broj\_pokretanja\_SelectSorta**,** broj\_usporedbi**;**

100

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

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

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

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

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

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

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

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

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

110 **// stack overflow.**

111

112 **Decimal32**

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

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

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

116

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

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

119

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

121 broj\_usporedbi **+=** **1;**

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

123 **EndIf;**

124 broj\_usporedbi **+=** **2;**

125 **If(**x **<** **8192** **and**

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

127 **// pojašnjenje ovoga...**

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

129 **Return**

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

131 **}**

132 **EndIf;**

133 broj\_usporedbi **+=** **1;**

134 **If** x **>** **8192** **Then** **{**

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

136 **}**

137 **EndIf;**

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

139 **// srednjoškolska matematika.**

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

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

142 broj\_usporedbi **+=** **4;**

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

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

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

146 **// čine nečitkima.**

147 **}**

148 **EndWhile;**

149 broj\_usporedbi **+=** **1;**

150 **If** x **<** **8192** **Then** **{**

151 **Return**

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

153 **:** **=** zbroj**;**

154 **}**

155 **EndIf;**

156 **Return** zbroj**;**

157 **}**

158 **EndFunction;**

159

160 **Decimal32**

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

162 **// sigurno ga neće usporiti...**

163

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

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

166 **// što se definira.**

167

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

169 broj\_usporedbi **+=** **2;**

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

171 **not(**memoizacija\_Eulerovog\_algoritma

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

173 **256]** **=** **0))**

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

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

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

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

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

179 **// compiler.**

180 **Return** memoizacija\_Eulerovog\_algoritma

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

182 **}**

183 **EndIf;**

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

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

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

187 broj\_usporedbi **+=** **4;**

188 y **+=** epsilon **\*** y**;**

189 i **+=** epsilon**;**

190 **}**

191 **EndWhile;**

192 broj\_usporedbi **+=** **1;**

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

194 **Return** memoizacija\_Eulerovog\_algoritma

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

196 **:** **=** y**;**

197 **}**

198 **EndIf;**

199 **Return** y**;**

200 **}**

201 **EndFunction;**

202

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

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

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

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

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

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

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

210 **// na desno nego na lijevo.**

211 broj\_usporedbi **+=** **1;**

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

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

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

215 **}**

216 **EndFunction;**

217

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

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

220 broj\_usporedbi **+=** **2;**

221 **If** DEBUG **=** **1** **Then** **{**

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

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

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

225 printFloat**(**x**);**

226 printFloat**(**y**);**

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

228 **}**

229 **EndIf;**

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

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

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

233 **}**

234 **EndIf;**

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

236 **// riskiram da će JavaScript**

237 **// virtualna mašina prekinuti**

238 **// izvođenje programa jer je**

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

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

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

242 **// milijarde u pozitivno i**

243 **// negativno).**

244 **}**

245 **EndFunction;**

246

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

248 **Decimal32** result

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

250 broj\_usporedbi **+=** **2;**

251 **Return** x **=**

252 **0** **?** **0** **:** ostatak\_pri\_dijeljenju**(**x**,** **2)** **=** **1** **and** x **<** **0** **?** **-**result **:** result**;**

253 **}**

254 **EndFunction;**

255

256 **Function** okreni\_dio\_niza**(Integer32** donja\_granica**,**

257 **Integer32** gornja\_granica**)** **Which** **Returns** **Nothing** **Does** **{**

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

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

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

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

262 j **-=** **1;**

263 i **+=** **1;**

264 broj\_usporedbi **+=** **1;**

265 **}**

266 **EndWhile;**

267 i**:**

268 **=** donja\_granica**;**

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

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

271 i **+=** **1;**

272 broj\_usporedbi **+=** **1;**

273 **}**

274 **EndWhile;**

275 **}**

276 **EndFunction;**

277

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

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

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

281 broj\_usporedbi **+=** **1;**

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

283 **// 2 (0 ili 1), znači da je već**

284 **// poredan, pa prekidamo**

285 **// izvođenje ovog potprograma.**

286 **Return;**

287 **}**

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

289 **// vrijedi ga posebno**

290 **// obraditi jer time**

291 **// možemo znatno ubrzati**

292 **// program.**

293 broj\_usporedbi **+=** **1;**

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

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

296 broj\_usporedbi **+=** **1;**

297 **Integer32** pomocna\_varijabla\_za\_zamijenu

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

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

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

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

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

303 **}**

304 **EndIf;**

305 **Return;**

306 **}**

307 **ElseIf** gornja\_granica **-**

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

309 **73** **Then** **{**

310 **// Za male je nizove SelectionSort brži i od MergeSorta i QuickSorta.**

311 **// Također, kako nije rekurzivan, može se koristiti i kad posve potrošimo**

312 **// memoriju na sistemskom stogu(na JavaScript Virtualnoj Mašini to jest ne**

313 **// više nego 4 KB, kako bijaše u doba Netscapea 2, godine 1996, tako i**

314 **// danas).**

315 broj\_usporedbi **+=** **2;**

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

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

318 broj\_usporedbi **+=** **2;**

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

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

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

322 **Break;**

323 **}**

324 **EndIf;**

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

326 **}**

327 **EndWhile;**

328 **If** je\_li\_niz\_vec\_poredan **Then** **{**

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

330 **Return;**

331 **}**

332 **EndIf;**

333 **Integer32** je\_li\_niz\_obrnuto\_poredan **:** **=** **1;**

334 i**:**

335 **=** donja\_granica**;**

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

337 broj\_usporedbi **+=** **2;**

338 **If** **ValueAt(**originalni\_niz **+** i**)** **<** **ValueAt(**originalni\_niz **+** i **+** **1)** **Then** **{**

339 je\_li\_niz\_obrnuto\_poredan**:**

340 **=** **0;**

341 **Break;**

342 **}**

343 **EndIf;**

344 i **+=** **1;**

345 **}**

346 **EndWhile;**

347 **If** je\_li\_niz\_obrnuto\_poredan **Then** **{**

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

349 okreni\_dio\_niza**(**donja\_granica**,** gornja\_granica**);**

350 **Return;**

351 **}**

352 **EndIf;**

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

354 i**:**

355 **=** donja\_granica**;**

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

357 broj\_usporedbi **+=** **1;**

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

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

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

361 broj\_usporedbi **+=** **2;**

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

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

364 gdje\_je\_minimum

365 **/\***

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

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

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

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

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

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

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

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

374 **\*/**

375 **:**

376 **=** j**;**

377 **}**

378 **EndIf;**

379 j **+=** **1;**

380 **}**

381 **EndWhile;**

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

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

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

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

386 i **+=** **1;**

387 **}**

388 **EndWhile;**

389 **Return;**

390 **}**

391 **EndIf;**

392 **Decimal32** razvrstanost **:** **=** **0;**

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

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

395 broj\_usporedbi **+=** **2;**

396 razvrstanost **+=**

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

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

399 broj\_usporedbi **+=** **1;**

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

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

402 **=** **0;**

403 **}**

404 **EndIf;**

405 i **+=** **1;**

406 **}**

407 **EndWhile;**

408 razvrstanost**:**

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

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

411 broj\_usporedbi **+=** **1;**

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

413 **// To ne smije biti...**

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

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

416 **// debuggerima za JavaScript**

417 **// virtualnu mašinu ako dođe**

418 **// do problema, lakše mi**

419 **// je ispisati brojeve u**

420 **// programu nego tražiti**

421 **// kako narediti**

422 **// debuggeru da ih**

423 **// ispiše.**

424 i**:**

425 **=** donja\_granica**;**

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

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

428 i **+=** **1;**

429 **}**

430 **EndWhile;**

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

432 **asm(**"unreachable"**);**

433 **}**

434 **EndIf;**

435 broj\_usporedbi **+=** **1;**

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

437 **// Opet ne smije biti...**

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

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

440 i**:**

441 **=** donja\_granica**;**

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

443 broj\_usporedbi **+=** **1;**

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

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

446 **}**

447 **EndWhile;**

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

449 **asm(**"unreachable"**);**

450 **}**

451 **EndIf;**

452 broj\_usporedbi **+=** **1;**

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

454 **// Open ne smije biti...**

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

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

457 i**:**

458 **=** donja\_granica**;**

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

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

461 i **+=** **1;**

462 **}**

463 **EndWhile;**

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

465 **asm(**"unreachable"**);**

466 **}**

467 **EndIf;**

468 **// Idemo dalje...**

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

470 **// izračunati ako ne**

471 **// pozivamo "pow" gdje**

472 **// ne treba (kad je**

473 **// eksponent prirodan**

474 **// broj).**

475i**:**

476 **=** **1;**

477 **While** i **<** **8** **Loop** **{**

478 broj\_usporedbi **+=** **1;**

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

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

481 i **+=** **1;**

482 **}**

483 **EndWhile;**

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

485 **// usporedbi QuickSort napraviti:**

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

487 **Decimal32** polinom\_pod\_apsolutnom

488 **:** **=** **2.38854** **\*** razvrstanost\_na\_potenciju**[7]** **-**

489 **0.284258** **\*** razvrstanost\_na\_potenciju**[6]** **-**

490 **1.87104** **\*** razvrstanost\_na\_potenciju**[5]** **+**

491 **0.372637** **\*** razvrstanost\_na\_potenciju**[4]** **+**

492 **0.167242** **\*** razvrstanost\_na\_potenciju**[3]** **-**

493 **0.0884977** **\*** razvrstanost\_na\_potenciju**[2]** **+** **0.315119** **\*** razvrstanost**;**

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

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

496 prirodni\_logaritam**(**

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

498 **1.05** **+**

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

500 prirodni\_logaritam**(**

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

502 prirodni\_logaritam**(2))** **\***

503 **0.9163** **\*** abs**(**polinom\_pod\_apsolutnom**);**

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

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

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

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

508 **(**gornja\_granica **-**

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

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

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

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

513 **// dijelova originalnog niza u**

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

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

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

517 **// da je cjelokupni algoritam brži**

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

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

520 **// pa ćemo preciznijim mjerenjima**

521 **// vidjeti što je bolje.**

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

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

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

525 broj\_usporedbi **+=** **1;**

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

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

528 **Return;**

529 **}**

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

531 broj\_usporedbi **+=** **1;**

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

533 okreni\_dio\_niza**(**donja\_granica**,** gornja\_granica**);**

534 **Return;**

535 **}**

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

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

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

539 prirodni\_logaritam**(2))**

540 **// JavaScriptina virtualna mašina ima**

541 **// 4KB memorije na sistemskom stogu,**

542 **// i alociranje više heap memorije**

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

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

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

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

547 **)** **and**

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

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

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

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

552 **Then** **{**

553 broj\_usporedbi **+=** **1** **+** **1** **+** **1** **+** **1;**

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

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

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

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

558 **// nije opcija)...**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

575 broj\_usporedbi **+=** **3;**

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

577 sredina\_niza **or**

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

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

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

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

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

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

584 **}**

585 **Else** **{**

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

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

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

589 **}**

590 **EndIf;**

591 i **+=** **1;**

592 **}**

593 **EndWhile;**

594 i**:**

595 **=** donja\_granica**;**

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

597 broj\_usporedbi **+=** **1;**

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

599 i **+=** **1;**

600 **}**

601 **EndWhile;**

602 **Return;**

603 **}**

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

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

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

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

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

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

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

611 **// poslije njega.**

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

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

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

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

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

617 broj\_usporedbi **+=** **2;**

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

619 i **+=** **1;**

620 pomocna\_varijabla\_za\_zamijenu**:**

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

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

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

624 **}**

625 **EndIf;**

626 j **+=** **1;**

627 **}**

628 **EndWhile;**

629 pomocna\_varijabla\_za\_zamijenu**:**

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

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

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

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

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

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

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

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

638 **Return;**

639 **}**

640 **EndIf;**

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

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

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

644 **asm(**"unreachable"**);**

645 **}**

646 **EndFunction;**

647

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

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

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

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

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

653 **}**

654 **EndIf;**

655 **// Testiraj matematičke funkcije...**

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

657 **// jako brze.**

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

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

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

661 **}**

662 **EndIf;**

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

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

665 velicina\_niza**:**

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

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

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

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

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

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

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

673 originalni\_niz**:**

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

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

676 pomocni\_niz**:**

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

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

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

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

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

682 **asm(**"unreachable"**);** **// Prekini izvršavanje ovog programa.**

683 **}**

684 **EndIf;**

685 **}**

686 **EndIf;**

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

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

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

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

691broj\_obrnuto\_poredanih\_podniza**:**

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

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

694 **:** **=** broj\_usporedbi

695 **:** **=** **0;** **// Nisam mogao odoljeti da u svoj programski jezik ne dodam**

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

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

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

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

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

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

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

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

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

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

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

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

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

709 izvijesti\_o\_broju\_usporedbi**(**broj\_usporedbi**);**

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

711 **Integer32** i **:** **=** **0;**

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

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

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

715 **asm(**"unreachable"**);**

716 **}**

717 **EndIf;**

718 i **+=** **1;**

719 **}**

720 **EndWhile;**

721 **}**

722 **EndFunction;**

723