Liet. matem. rink., T. 40, spec. nr., 184–190

Vardų sintaksė ir mnemonika programavimo kalbose

Mikas GRIGALIŪNAS, Gintautas GRIGAS (MII)

el. paštas: mikas.grigaliunas@maf.vu.lt, grigas@ktl.mii.lt

1. Įvadas

Esminė vardų savybė programose yra jų mnemonika – vardas turi atspindėti juo pažymėto objekto prasmę. Gerai parinkti vardai padeda suvokti, ką jie žymi, lengva juos įsiminti. Perėjimas nuo beprasmių skaitmeninių adresų prie prasmingų vardų buvo esminis lūžis programavime, įvykęs praėjusio šimtmečio viduryje.

Vardų sudarymo taisykles apibrėžia programavimo kalba. Nuo jų priklauso potencialios vardų mnemonikos galimybės.

Pagal nusistovėjusias taisykles vardai sudaromi iš raidžių ir skaitmenų:

<pirmasis vardo simbolis> ::= <raidė>

<tolesnis vardo simbolis> ::= <raidė> | <skaitmuo>

<vardas> ::= <pirmasis vardo simbolis> (<tolesnis vardo simbolis>)*

Lemiamą įtaką vardų mnemonikai turi raidžių, kurias galima vartoti varduose, aibė, kurios dydis priklauso nuo kodavimo – bitų skaičiaus vienam ženklui.

6 bitų kodavimas. Buvo naudojamas pirmuose kompiuteriuose. Į 6 bitų abėcėlę tilpo tik vieno registro (tik didžiosios arba tik mažosios) 26 pagrindinės lotyniškosios abėcėlės raidės. Kompiuteriai su 6 bitų kodavimu seniai nebenaudojami. Tuo tarpu 6 bitų kodavimo reliktai dar yra išlikę programavimo kalbose, kurių vardų sudarymo taisyklės leidžia naudoti tik 26 lotyniškosios abėcėlės raides, o didžiąsias ir mažąsias raides laiko lygiavertėmis.

7 bitų kodavimas. Dvigubai padidėjo ženklų skaičius, o tuo pačiu ir raidžių – atsiskyrė mažosios raidės nuo didžiųjų. Šį kodavimą atspindi programavimo kalbos, kuriose didžiosios ir mažosios raidės laikomos skirtingomis, bet raidžių aibė ribojama 26 pagrindinėmis lotyniškosios abėcėlės raidėmis. Šitokia raidžių aibė visiškai patenkino anglų kalbą ir todėl įsigalėjo programavimo kalbose.

8 bitų kodavimas. Raidžių abėcėlė praplėsta įvairių kalbų raidėmis. Todėl atsirado galimybė programose vartoti tikrus gyvenimiškus ne tik anglų kalbos, bet ir kitų kalbų vardus. Tačiau 8 bitų dar nepakako hieroglifus vartojančiose kalbose.

16 bitų kodavimas (Unikodas). Tai universalus kodavimas, apimantis

2. Bylų ir katalogų vardai operacinėse sistemose

Programavimo kalbos gyvuoja ilgiau, negu kompiuteriai. Dar intensyviai tebenaudojamas Paskalis ir C, bei įvairios jų modifikacijos, nors šioms kalboms jau apie 30 metų. Todėl iškyla problema, kaip šias kalbas suderinti su šiuolaikine skaičiavimo technika.

Operacinės sistemos glaudžiau siejasi su kompiuteriu ir dėl to jose greičiau panaudojamos kompiuteriuose atsiradusios naujovės. Todėl verta pažvelgti į situaciją su vardais operacinėse sistemose ir pasinaudoti čia sukaupta patirtimi sprendžiant analogiškas problemas programavimo kalbose.

Jau nuo DOS 3.3 versijos bylų ir katalogų varduose buvo galima vartoti visas raides tos kodų lentelės, kuri naudojama operacinėje sistemoje. Taip pat pasirūpinta įvairiapusišku vardų, kuriuose naudojamos visos raidės, apdorojimu: bylų ir katalogų rikiavimu pagal abėcėlę, didžiųjų ir mažųjų raidžių ekvivalentumu. Taigi, raidžių aibė jau turtinga ir išnaudojanti 8 bitų kodavimo galimybes. Tuo tarpu dar nėra skirtumo tarp didžiųjų ir mažųjų raidžių (nepilnai išnaudota 7, o tuo pačiu ir 8 bitų kodavimo galimybė). Tai rodo, kad turėti tinkama raidžių aibę yra svarbiau, negu du raidžių lygius (mažąsias ir didžiąsias raides).

Operacinėse sistemose DOS ir „Windows" vartojamos skirtingos kodų lentelės. Tačiau bylų ir katalogų vardai abiejose sistemose yra suderinti ir tie patys vardai rodomi teisingai abiejose sistemose, nežiūrint kurioje jų jie būtų sukurti.

3. Kalbų koregavimas

Programavimo kalbų semantika neapibrėžia raidžių prasmės. Raidės tarnauja tik kaip „statybinė medžiaga" sudaryti vardams, kurie jau turi semantiką, bet nepriklausomą nuo vardą sudarančių simbolių. Dėl to teoriškai raidžių aibė programavimo kalbai nėra svarbi ir ją galima parinkti tokią, kokia yra kompiuteryje.

Greičiausiai prie šiuolaikinių kompiuterių su 8 bitų kodavimu prisitaikė Logo šeimos kalbos. Programų varduose vartojamos visos tos kalbos raidės, kuriai parašytas arba į kurią išverstas transliatorius. Turime dvi Logo sistemas, kuriose dirbama su visomis lietuviškos abėcėlės raidėmis – Logo Writer [4] ir Komenskio Logo. Tuo tarpu daugumoje profesionalių programavimo kalbų dar tebėra 26 raidžių ribojimas. Tai galima paaiškinti tuo, kad jose stengiamasi išlaikyti suderinamumą su ankstesnėmis tų kalbų versijomis. Tačiau padėtis keičiasi.

Tam, kad būtų galima turėti pilnavertę abėcėlę nenusižengiant galiojantiems kalbos standartams, į naujas programavimo kalbų (Paskalio [1,2], Modulos-2 [5], Ados 95 [6] ir kt.) standartų redakcijas įtraukiamas punktas, teigiantis, jog konkrečioje realizacijoje (transliatoriuje), simbolių aibę galima papildyti ir bus laikoma, kad realizacija suderinta su standartu. Reikia tik tokius papildymus apibrėžti kalbos (transliatoriaus) dokumentacijoje. Dar daugiau, leidžiama kalbą papildyti ne tik raidėmis, bet ir reikšmingais simboliais, t.y. tokiais, kurie turi įtakos kalbos semantikai, pavyzdžiui, baziniais žodžiais.

Reikia pažymėti, kad kalbų variantuose bei transliatoriuose ženklų, vartojamų varduose, aibės praplėtimas nėra naujiena. Pavyzdys – beveik visuose Paskalio kalbos transliatoriuose ši aibė yra papildyta pabraukimo ženklu, kurio nėra Paskalio kalbos standarte [2].

4. Turbo Paskalio abėcėlės papildymas lietuviškomis ir latviškomis raidėmis

Turbo Paskalis yra vienas seniausių ir dar dabar tebenaudojamų kompiliatorių, kuris nuo 1992 m. nebeatnaujinamas dėl vidinių jį sukūrusios korporacijos (Borland, vėliau Inprise) problemų. Dėl to atsirado poreikis išorinėmis priemonėmis jį papildyti taip, kad juo būtų galima naudotis tiesiogiai iš „Windows 95" ir vėlesnių šios sistemos versijų terpės. Kadangi DOS ir „Windows" terpėse vartojamos skirtingos kodų lentelės, tai reikalingas perkodavimas, o kartu su perkodavimu buvo galima realizuoti ir vardų koregavimą.

Šiai idėjai realizuoti buvo sudaryta sąsaja Turbo Paskalį paleidžianti iš tekstų redaktoriaus „MS Word 97" ir naujesnės jo versijos „MS Word 2000". Programos tekstas rašomas „MS Word" redaktoriumi. Varduose galima naudoti visas lietuviškas ir latviškas raides. Kai programa pateikiama transliatoriui, raidės, turinčios diakritinių ženklų, pakeičiamos ženklų _ (pabraukimo brūkšnys) ir jas atitinkančių pagrindinės lotyniškosios abėcėlės raidžių sekomis. Vienas pabraukimo brūkšnys žymi raides, turinčias vieną modifikaciją su diakritiniu ženklu. Abiejų kalbų abėcėlėse yra trys raidžių poros, kilusios iš tų pačių pagrindinių lotyniškosios abėcėlės raidžių (Ą ir Ā iš A, Į ir Ī iš I, Ų ir Ū iš U). Poros raidė, turinti diakritinį ženklą žemiau raidės (Ą, Į ir Ų) žymima vienu pabraukimo brūkšniu, o virš raidės (Ā, Ī, Ū) – dviem. Raidžių trejeto (Ę, Ė ir Ē iš E) trečiajai raidei koduoti teko panaudoti tris brūkšnius. O pats pabraukimo brūkšnys, jeigu jis panaudojamas varde, pakeičiamas ženklais b_ (toks pakeitimas galimas, nes raidės b su diakritiniais ženklais nėra).

Toks kodavimas nėra universalus ir ribotesnis negu, pavyzdžiui, kodavimas „Quoted Printable", vartojamas elektroniniame pašte. Tačiau jis lengviau skaitomas, kai žmogui tenka tiesiogiai skaityti perkoduotą tekstą, pavyzdžiui, kai jis perkeliamas į kitą Paskalio transliatorių, neturintį čia aprašytos sąsajos.

Kaip pasinaudoti šia sąsaja, yra aprašyta straipsnyje [7].

Šis sprendimas nepretenduoja į universalumą, o yra tik pavyzdys, kaip galima seną programinę įrangą pritaikyti naujomis sąlygomis.

5. Komponentinis Paskalis

Aštuonių bitų kodavimą išnaudoja 1997 m. sukurtas Komponentinis Paskalis. Jo raidžių aibę sudaro pagrindinės 26×2 lotyniškosios abėcėlės raidės ir 31×2 kitos lotyniškos abėcėlės raidės (su diakritiniais ženklais) kurių kodai yra iš intervalo nuo 192 iki 255, išskyrus dvi pozicijas, kurias užima du simboliai × ir ÷, įsiterpę tarp raidžių. Taip yra ir „gimtojoje" Komponentinio Paskalio sistemoje „Black Box".

6. Unikodas Javoje

Dabar jau einama prie Unikodo. Čia vienam simboliui koduoti skiriama 16 bitų ir į tokį kodą telpa visų pasaulio kalbų raidės. Raidžių daug. Visų jų programavimo kalbos aprašyme neišvardysi. Todėl iškyla problema, kaip tokią aibę apibrėžti programavimo kalbos sintaksėje. Javoje einama paprasčiausiu keliu. Vardams sudaryti gali būti vartojamos 26 pagrindinės didžiosios ir mažosios lotyniškos abėcėlės raidės, o taip pat visi simboliai, kurių kodai Unikode yra didesni už U+0080 (arba už dešimtainį 160). Tačiau čia yra ne tik raidės, bet ir daugybė kitokių ženklų (matematikos, grafikos ir įvairių kitokių). O jie nebūdingi vardams ir užuot palengvinę programos skaitymą, gali įnešti ir nereikalingos painiavos. Taigi sprendimas gana primityvus ir perdaug bendras.

7. Unikodas Adoje

Adoje priimtas racionalesnis sprendimas, negu Javoje. Ados 95 [6] standarte pasakyta, kad varduose vartojamos visos lotyniškosios abėcėlės raidės. Jų sąrašas būtų ilgas, todėl raidės nevardijamos, o apibrėžiama, kad raide laikomas simbolis, kurio pavadinimas ISO/IEC 100646 standarto (Unikodo) angliškame variante prasideda žodžiais LATIN CAPITAL LETTER arba LATIN SMALL LETTER. Taigi vardai **ženklas, Sigurðsson** yra taisyklingi. Tuo tarpu vardai **имя**, φ – netaisyklingi, kadangi juose vartojamos ne lotyniškos abėcėlės raidės.

Raidėmis iš kitokių abėcėlių (pvz., kirilicos, graikų, arabų, hebrajų, kinų) Adą 95 galima tik papildyti anksčiau minėtu būdu.

Raidžių aibės nustatymas pagal raidžių pavadinimus teoriškai aiškus, bet kaip jį realizuoti transliatoriuje?

Su panašiomis problemomis susiduria ir daugelis kitų tekstus apdorojančių programų. Todėl jos sprendžiamos sistemingai: Unikodas pateikia lenteles su užkoduotomis simbolių savybėmis. Belieka tik jomis pasinaudoti.

8. Universalios vardų sudarymo taisyklės Unikode

Norint išlaikyti vienodą vardų sudarymo sistemą įvairiose programavimo kalbose, reikia bendrų susitarimų. Tuo pasirūpinta pačiame Unikode. Jame apibrėžtos vardų sintaksės taisyklės [3].

Pirmiausia apibūdinsime aibes simbolių, vartojamų vardams sudaryti:

- abėcėlės simboliai – tai bet kurios kalbos raidės, dviraidžiai, ligatūros bei jų kontekstiniai variantai, raidžių modifikatoriai, kombinuojamieji ir kiti simboliai;
- kombinuojamieji simboliai – tai abėcėlės simbolių poaibis. Šie simboliai yra vietos neužimantys diakritiniai ženklai, vartojami raidėms, kurių nėra Unikode, išreikšti, pavyzdžiui, raidė Ą su riestiniu kirčiu išreiškiama raidės Ą ir kombinuojamos tildės ~ pora Ą~;

- pradiniai abėcėlės simboliai – abėcėlės simboliai be kombinuojamųjų simbolių;
- hieroglifai;
- dešimtainiai skaitmenys;
- gaubiamieji simboliai ={20DD, 20DE, 20DF, 20E0};
- nulinio pločio simboliai ={200C, 200D, 200E, 200F};
- rašymo krypties simboliai ={202A, 202B, 202C, 202D, 202E};
- formatavimo simboliai ={206A, 206B, 206C, 206D, 206E, 206F};
- jungiamasis tarpas ={FEFF};
- pabraukimo simboliai ={005F,FF3F};
- papildomi simboliai ={00B7, 02D0, 02D1, 0387, 0640, 0E46, 0EC6, 3005, 3031..3035, 309B..309D, 309E, 30FC..30FE, FF70, FF9E, FF9F}.

Iš šių simbolių aibių vardai sudaromi pagal tokias taisykles:

<vardo kombinuojamasis simbolis> ::=
<kombinuojamasis simbolis> – <gaubiamasis simbolis>
<varde ignoruojamas simbolis> ::= <nulinio pločio simbolis> |
<rašymo krypties simbolis> |
<formatavimo simbolis> | <jungiamasis tarpas>
<pirmasis vardo simbolis> ::= <pradinis abėcėlės simbolis> | <hieroglifas>
<tolesnis vardo simbolis> ::= <abėcėlės simbolis> | <hieroglifas> | <skaitmuo> |
<vardo kombinuojamasis simbolis> |
<pabraukimo simbolis> |
<papildomas simbolis> |
<varde ignoruojamas simbolis>
<vardas> ::= <pirmasis vardo simbolis> (<tolesnis vardo simbolis>)*

Taisyklėse minuso ženklu pažymėtas aibių skirtumas.

Taisyklingų vardų pavyzdžiai:

Temperatūra°C

pólnoc

φ

$B\alpha\beta\nu\lambda\acute{\omega}\nu$

Буква

Ɔ

Simbolių gausa ir įvairovė atveria naujas galimybes programų rašytojams. Bet ar nesukels problemų transliatorių autoriams?

Didelių problemų neturėtų būti, kadangi čia panaudojamos simbolius klasifikuojančios savybės, formaliai ir vienareikšmiškai gaunamos iš minėtų Unikodo lentelių, kurias jau turi operacinės sistemos, nes tokios lentelės reikalingos daugeliui programų, apdorojančių Unikodo tekstus.

9. Išvados

1. Praktikoje dar tebenaudojamos programavimo kalbos su šiuolaikinių kompiuterių neatitinkančiais varduose naudotinos raidžių aibės ribojimais. Programavimo kalbų standartai nedraudžia raidžių aibes papildyti ir laiko, kad tokie papildymai yra suderinami su standartais.
2. Visų abėcėlės raidžių naudojimo problemos operacinių sistemų bylų ir katalogų varduose yra išspręstos.
3. Yra pateikta Turbo Paskalio ir „MS Word" sąsaja, įgalinanti naudoti visas lietuviškos ir latviškos abėcėlių raides Paskalio kalbos varduose.
4. Yra nedaug programavimo kalbų, naudojančių 8 bitų kodavimą, kurių varduose galima vartoti visas abėcėlės raides (Logo Writer, Komenskio Logo, Komponentinis Paskalis).
5. Programavimų kalbų, naudojančių Unikodą (Ada, Java), varduose galima naudoti visas abėcėlės raides. Raidžių aibė apibrėžiama įvairiai.
6. Unikode yra pateiktos universalios vardų sudarymo programavimo kalbose sintaksės taisyklės.
7. Pirmasis lūžis vardų mnemonikoje įvyko prieš pusę šimtmečio, kai vietoj skaitmeninių adresų pradėta vartoti raidinius žymėjimus. Antrasis lūžis vyksta dabar, kai vietoj vardų, sudarytų iš ribotos abėcėlės raidžių, imami vartoti laisvai pasirinkti vardai, kuriuose gali būti pavartotos visos abėcėlės raidės.

Literatūra

[1] Extended Pascal, ISO/IEC 10206:1990.
[2] Pascal. ISO/IEC 7185:1990.
[3] The Unicode Standard, Version 2.0, Addison-Wesley Developers Press (1993).
[4] A. Balvočienė, V. Dagienė, A. Klupšaitė, *Logo žinynas*, Vilnius, Folium (1996).
[5] *Modula-2*, ISO/IEC 10514-1:1996.
[6] *Ada 95*, ISO/IEC 8652:1995.
[7] M. Grigaliūnas, Į Paskalio programą – iš „Word", *Kompiuterija*, 3, p.40 (2000).

Identifier syntax and mnemonics in programming languages

M. Grigaliūnas, G. Grigas

The main attention in the paper is paid to the set of symbols, which can be used in identifiers. Modern computers uses 8 bit coding, and many programming languages does so. But only few of them (Logo, Component Pascal) takes all advantages of 8 bit coding and allows to use all letters of alphabet in identifiers. Proffesional languages such as Pascal and C have restrictions for a symbol set in identifiers. These restrictions comes back from 7 or even 6 bit coding. The recent standards of programming languages allows to extend symbol set. Such kind of extension – interface between TurboPascal and MS Word, which allows to use Lithuanian and Latvian letters in Pascal program identifiers, is presented in this paper. Some languages (Ada95, Java) already uses Unicode. The sets of symbols for identifiers in these languages are defined in different ways. Unicode itself provides syntax rules for identifiers, in order to ensure uniform treatment of identifiers.