Sisällysluettelo

1 Tehtävän määritys

Tehtävänä oli tehdä ohjelma, joka muuttaa annetun kokonaisluvun väliltä $0 – 10^7$ sanalliseen muotoon. Alkuperäinen tehtävän määritys on Liittessä 1.

2 Käyttöohjeet

2.1 Kääntäjät

Kääntäjinä käytettiin Borland C++ -kääntäjää (v5.6), GNU C-kääntäjää (v.3.4.1 Cygwin) ja Microsoftin C++-kääntäjää (.Net 2003 13.x). Ohjelman kohdeympäristo on Windows 98/NT/XP PC. Käännetty ohjelma (LUKUSANA.EXE) on siis Win32-sovellus, joka voidaan suorittaa vain Windowsin ollessa ladattuna. Lopullinen versio ohjelmasta on käännetty GNU C-kääntäjällä, ja ohjelma tarvitsee cygwin1.dll -tiedoston. Jos ohjelma käännetään vanhalla DOS:in C-kääntäjällä, ääkköset tulostuvat todennäköisesti väärin.

2.2 Käyttö

Ohjelma käynnistetään komentoriviltä DOS-ikkunassa antamalla ohjelman nimi ja sen perään luku, joka halutaan muuntaa:

```
LUKUSANA.EXE 123
```

Ohjelman tuloste:

```
Ohjelma muuttaa annetun luvun sanalliseen muotoon.
Annoit luvun: 123.
Antamasi luku sanallisessa muodossa: satakaksikymmentäkolme
```

Ohjelma hyväksyy luvut väliltä 0 – 10000000, ja syötteessä ei saa muita merkkejä kuin numeroita. Ei edes etumerkkiä tai desimaalipistettä tai -pilkkua. Tulostuksen jälkeen ohjelman suoritus loppuu. Tulosteiden yhteenkirjoitus noudattaa suomen kielioppia..

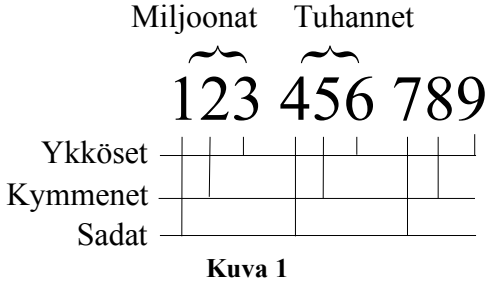
3 Ratkaisun kuvaus

3.1 Ohjelman toiminta yleisesti

Aluksi ohjelma tarkistaa, onko se käynnistetty oikealla parametrien lukumäärällä. Oikea lukumäärä on 2 (ohjelman nimi + syöte). Jos parametreja on liikaa tai ei ollenkaan, tulostetaan käyttöohjeet. Jos parametreja on oikea määrä, tarkistetaan syöte. Tutkitaan, että se koostuu vain numeromerkeistä ja se pystytään muuntamaan kokonaisluvuksi. Jos luvuksi muunto onnistui, tarkistetaan, että luku on sallituissa rajoissa. Hyväksytylle luvulle tehdään muunnos sanalliseen muotoon ja tulostetaan lopuksi..

3.2 Muunnoksen periaate

Kokonaisluku jaetaan oikealta vasemmalle kolmen numeron ryhmiin. Ryhmissä toistuu sama jako satoihin, kymmeniin ja ykkösiin (Kuva 1). Kukin ryhmä voidaan muuntaa sanalliseen muotoon riippumatta toisista ryhmistä. Kunkin ryhmän sanallisen esityksen jälkeen tarvitsee vain tulostaa, onko kyse miljoonista, tuhansista vai ykkösistä (ykkösille ei tulostu tekstiä).



Kuva 1

Koska tulostus käyttää samoja tekstejä koko ajan, ne on toteutettu tekstivakioilla. Ainoa muuttuva asia on niiden keskinäinen tulostusjärjestys; sitä varten tarvitaan oma tietorakenne (pino).

3.3 Muunnosalgoritmi

Algoritmin syöte on ei-negatiivinen kokonaisluku (= LUKU). Tulosteet menevät oletettuun pinoon.

1. Jos LUKU on nolla, tallenna tuloste ”nolla”. Suoritus loppuu.
2. Nollaa tuhansien eksponenttilaskuri.
3. Ota luvusta LUKU oikealta 3 viimeistä numeroa omaksi luvukseen (= LUKU3).
4. Jos tämä LUKU3 on nolla, siirry kohtaan 12.
5. Jos tuhansien eksponentti on vähintään 1, tutki LUKU3:n arvo. Jos se on yksi, tallenna eksponentin mukainen tuloste ”tuhat”, ”miljoona”, ”miljardi” ja siirry kohtaan 12. Muutoin tallenna eksponentin mukainen tuloste ”tuhatta”, ”miljoonaa”, ”miljardia”.
6. Ota luvusta oikealta 2 viimeistä numeroa omaksi lukuarvoksi (= LUKU2).
7. Jos LUKU2 on välillä 0 - 19, tallennetaan vastaava tuloste ”yksi”...”yhdeksäntoista”. ”nolla” hypätään yli, se on jo käsitelty. Siirry kohtaan 11.
8. Tutki LUKU3:n viimeinen numero ja tallenna sen mukaan tuloste ”yksi”...”yhdeksän”. ”nolla” hypätään yli.

9. Tallenna tuloste ”kymmentä”.
10. Tutki LUKU3:n keskimmäinen numero ja tallenna sen mukainen tuloste ”kaksi”...”yhdeksän”.
11. Tutki LUKU3:n ensimmäinen numero. Jos se on yksi, tallenna ”sata”. Muutoin tallenna tuloste ”sataa” ja sen jälkeen tallenna 1. numeron mukainen tuloste ”kaksi”..."yhdeksän”.
12. Jaa LUKU 1000:lla ja sijoita osamäärä LUKU :uun takaisin. Jos LUKU on nolla, suoritus loppuu. Muutoin kasvata tuhansien eksponenttia yhdellä ja siirry kohtaan 3.

3.4 Vakiot

Symboliset esikääntäjävakiot on listattu Taulukko 1:ssä.

Vakio	***Selite***	***Arvo***
LOWER_LIMIT	Pienin syötteeksi kelpaava luku	0
UPPER_LIMIT	Suurin syötteeksi kelpaava luku	10000000
ARG_PROGRAM_NAME	Ohjelma 0:s komentoriviparametri eli ohjelma itse	argv[0]
ARG_USER_INPUT	Ohjelman 1. komentoriviparametri	argv[1]
INPUT_OK	Syötteen tarkistava funktion paluuarvo, kun syöte on kokonaan hyväksytty	0
INPUT_ERROR	Syötteen tarkistava funktion paluuarvo, jos syöte ei joltakin osin kelpaa	-1
INPUT_PARAM_ERROR	Syötteen tarkistava funktion paluuarvo, jos jokin parametri on NULL	-2

Taulukko 1

Vakiotietorakenteet on kuvattu Taulukko 2:ssa. Ne ovat kaikki merkkijonovakioita.

Vakio	***Selite***	***Arvo / Arvot***
NUMBERS_0_19	Numerot 0 – 19. Taulukko.	”nolla” ... ”yhdeksäntoista”
TENS	Monikkoteksti kymmenille	”kymmentä”
HUNDREDS	Monikkoteksti sadoille	”sataa”
ONE_HUNDRED	Yksikköteksti sadoille	”sata”
THOUSANDS_BY_EXP	Monikkoteksti tuhansien eri potensseille. Taulukko.	”tuhatta”, ”miljoonaa”, ”miljardia”
THOUSAND_BY_EXP	Yksikköteksti tuhansien eri potensseille. Taulukko.	”tuhat”, ”miljoona”, ”miljardi”
SPACE	Välilyönti	” ”

Taulukko 2

3.5 Tietotyypit

3.5.1 Node

Luvun sanallinen muoto tallennetaan solmuihin. Solmun tietorakenne on tietue Node:

```
typedef struct node
    {
    struct node *next;      /* seuraavan solmun osoite */
    const char  *data;      /* itse tieto; osoite
vakiomerkkijonoon */
    } Node;
```

3.5.2 Stack

Solmut muodostavat linkitetyn pinon. Pinon tietorakenne on tietue Stack:

```
typedef struct
    {
    int   count; /* solmujen lukumäärä */
    Node  *top;  /* linkki päällimmäiseen solmuun */
    } Stack;
```

Tiedon tallennus tapahtuu lisäämällä uusi solmu pino päällimmäiseksi. Tiedon lukeminen kohdistuu pinon päällimmäiseen solmuun, joka sitten poistetaan.

3.6 Aliohjelmat

3.6.1 validate_and_convert_input

int **validate_and_convert_input**(const char ***input**, unsigned long ***number**)

Parametrit: input Osoitin käyttäjän antamaan syötteeseen

number Osoitin kokonaislukuun, johon syötteen numeerinen esitys tallennetaan

Paluuarvot: INPUT_OK Syöte hyväksytty ja muunnettu numeromuotoon

INPUT_ERROR Syötettä ei voitu muuntaa numeeriseen muotoon tai numeerinen muoto ei ole sallituissa rajoissa (0 – 10000000).

INPUT_PARAM_ERROR Jompikumpi parametri on NULL

Kuvaus: Aliohjelma yrittää aluksi muuntaa merkkijonona annetun syötteen numeeriseksi. Jos muunnos onnistuu, tarkistetaan vielä, että numeroarvo on välillä 0 – 10000000. Jos numeroarvo on hyväksytty, paluuarvoksi asetetaan INPUT_OK ja kutsuva koodi voi käyttää numeroarvoa.

3.6.2 pop_from_stack

const char ***pop_from_stack**(Stack ***stack**)

Parametrit: stack Osoite pinoon, johon operaatio kohdistuu

Paluuarvo: Osoite tekstivakioon tai NULL, jos pino oli tyhjä

Kuvaus: Aliohjelma palauttaa pinon päällimmäisenä olevan tekstivakion osoitteen. Jos pino on tyhjä, palautetaan NULL.

3.6.3 push_to_stack

void **push_to_stack**(Stack ***stack**, const char ***data**)

Parametrit: stack Osoite pinoon, johon operaatio kohdistuu

data Osoite tekstivakioon, joka tallennetaan pinoon

Paluuarvo: Ei mitään

Kuvaus: Aliohjelma asettaa annetun tekstivakion osoitteen pinoon päällimmäiseksi.

3.6.4 print_stack

void **print_stack**(Stack ***stack**)

Parametrit: stack Osoite pinoon, johon operaatio kohdistuu

Paluuarvo: Ei mitään

Kuvaus: Aliohjelma tulostaa pinon sisältämät tekstit peräkkäin näytölle. Pino tyhjenee.

3.6.5 create_stack

Stack ***create_stack**(void)

Parametrit: Ei mitään

Paluuarvo: Osoite luotuun pinoon

Kuvaus: Aliohjelma luo ja alustaa pinon.

3.6.6 destroy_stack

void **destroy_stack**(Stack ***stack**)

Parametrit: stack Osoite pinoon, johon operaatio kohdistuu

Paluuarvo: Ei mitään

Kuvaus: Aliohjelma poistaa pinon varaamaan muistin. Pinon on oltava tyhjä.

3.6.7 make_string_presentation

void **make_string_presentation**(unsigned long **number**, Stack ***stack**)

Parametrit: number Luku, joka halutaan esittää sanallisessa muodossa

stack Osoite pinoon, johon sanallinen muoto tallennetaan

Paluuarvo: Ei mitään

Kuvaus: Aliohjelma muuntaa annetun luvun sanalliseen muotoon. Tekstit tallennetaan pinoon, joka voidaan tulostaa omalla aliohjelmallaan.

4 Testaus

Testiajot on koottu liitteeseen 2. Testisyötteet on laadittu ohjetta noudattaen siten, että ne kattavat suurimman osan koodista ja tekstivakioista.

5 Yhteenveto

Keskeisin haaste oli jalostaa muunnosalgoritmi ensimmäisestä ideasta lopulliseen muotoonsa. Alussa tuli tutkittua, voisiko muunnosalgoritmi toimia suoraan merkkijonolla, mutta kätevintä oli muuntaa syöte numeeriseksi.Yllätyksen aiheutti strtoul-kirjastofunktion epäyhtenäinen toiminta unsigned long:in ylärajalla Borlandin ja GNU:n kääntäjien kesken. Osoittautui hyväksi ideaksi käyttää useaa kääntäjää ja verrata tuloksia. Lopullisen varmistuksen ohjelman ja algoritmin toimivuudesta sai Microsoftin kääntäjällä.

Toinen, pienempi tehtävä oli sopivan tallennusmuodon valinta algoritmin tuloksille. Pino olikin sopiva valinta, koska tulostuksen piti olla käänteinen verrattuna tekstin luontiin algoritmissa. Muut ohjelman toiminnot sisälsivät tyypillistä syötteen lukemista, tarkistusta ja tulostusta. Näissä ei ollut sen kummempaa ongelmaa kuin tulosteiden ääkkösten korjaus eri kääntäjille.

6 Liitteet

Liite 1: Tehtävän kuvaus

Liite 2: Testit

Liite 3: Lähdekoodi