Aplicație pentru conversii și calcule în diferite baze de numerație

# Documentație

# Enunțul problemei

Aplicaţia trebuie să exemplifice cele trei metode de conversie ale numerelor naturale (împărţiri succesive, substituţie şi utilizarea unei baze intermediare - baza 10) între două baze de numeraţie diferite de 10, conversiile rapide între bazele puteri ale lui 2 (2, 4, 8, 16) şi operaţiile aritmetice într-o bază oarecare p (adunare, scădere, înmulţire cu o cifră şi împărţire la o cifră), fără a trece numărul prin baza 10 (p din {2,3,...,9,10,16}). Aplicația va include deci 2 interfațe diferite accesibile dintr-un meniu, corespunzătoare celor 2 funcționalități: conversie, respective calcule.

Interfața de conversii permite introducerea unui număr întreg, respectiv selectarea unei baze inițiale și a unei baze finale. Utilizatorul poate alege, prin intermediul a 2 butoane diferite, modalitatea prin care se va efectua conversia (fie direct, fie printr-o bază intermediară). În urma apăsării butonului, se va afișa fie rezultatul conversiei, fie o eroare dacă este cazul (dacă numărul/bazele nu au fost introduse corect).

Interfața de calcule permite alegerea unei baze în care se vor efectua calculele, respectiv introducerea a 2 valori întregi între care se va efectua operația dorită. Din nou, la apăsarea unui buton va fi afișat, după caz, fie rezultatul sau textul erorii.

Aplicația va fi implementată pe baza principiilor de programare orientată cu obiecte în limbajul C#, pentru a obține o aplicație cât mai intuitivă și profesională din punct de vedere al designului. Algoritmii de calcul și conversie vor fi implementați în module Python care vor fi apelate ca procese prin linia de comandă, astfel încât operațiile să fie cât mai eficiente, iar codul cât mai simplu și inteligibil, organizat după principiul stratificării, pe clase. Modulele de tip Service vor coordona funcționalitățile aferente, prin accesul la librăria de algoritmi. Modulul librărie instanțiază entitățile de tip Număr și apelează algoritmii doriți din cadrul acestora.

# Diagrama de apel

GUI

Service conversii

Service conversii cu bază intermediară

Service calcule

Librărie algoritmi

Clasa număr

Validator

# Tipuri de date folosite

La nivelul modului GUI:

* numar: string
* baza: string
* operatie: string

La nivelul modulului Service:

* numar: integer
* baza: integer
* operatie: string

La nivelul clasei Număr:

* numar: lista de string
* baza: integer
* operatie: string

# Subalgoritmii principali

## Adunarea a 2 numere

* *Date de intrare*: op1, op2 (liste de cifre), baza
* *Date de ieșire*: suma (listă de cifre)
* *Precondiții*: -
* *Postcondiții*: -
* *Pseudocod*:

rezultat <-- []

t <-- 0

**cât timp** lungimea op1 != lungimea op2:

**dacă** lungimea op1 < lungimea op2:

**atunci**: completează op1 cu 0 la început

**altfel**: completează op2 cu 0 la început

**sfârșit dacă**

**sfârșit cât timp**

**pentru** i **de la** lungimea op1 **la** -1:

rezultat <-- rezultat + (op1[i]+op2[i]+t)%baza

t <-- (op1[i]+op2[i]+t)//baza

**sfârșit pentru**

**dacă** t != 0:

**atunci**: rezultat <-- rezultat + t

**sfârșit dacă**

inversează rezultatul

return rezultat

## Scăderea a 2 numere

* *Date de intrare*: op1, op2 (liste de cifre), baza
* *Date de ieșire*: diferența (listă de cifre)
* *Precondiții*: op1>=op2
* *Postcondiții*: -
* *Pseudocod*:

**dacă** lungimea op1 < lungimea op2:

**atunci**: eroare

**altfel**: **dacă** (lungimea op1 == lungimea op2) și (op1[0]<op2[0]):

**atunci**: eroare

**altfel:**

**cât timp** lungimea op1 != lungimea op2:

completează op2 cu 0 la început

**sfârșit cât timp**

rezultat <-- []

t <-- 0

**pentru** i **de la** lungimea op1 **la** -1:

**dacă** op1[i]+t >= op2[i]:

**atunci**: rezultat <-- rezultat + op1[i]+t-op2[i]

t <-- 0

**altfel**: rezultat <-- rezultat + baza+op1[i]+t-op2[i]

t <-- -1

**sfârșit dacă**

**sfârșit pentru**

inversează rezultatul

return rezultat

**sfârșit** **dacă**

**sfârșit dacă**

## Înmulțirea unui număr cu o cifră

* *Date de intrare*: op1 (listă de cifre), op2 (cifră), baza
* *Date de ieșire*: produsul (listă de cifre)
* *Precondiții*: -
* *Postcondiții*: -
* *Pseudocod*:

rezultat <-- []

t <-- 0

**pentru** i **de la** lungimea op1 **la** -1:

rezultat <-- rezultat + (op1[i]\*op2+t)%baza

t <-- (op1[i]\*op2+t)//baza

**sfârșit pentru**

**dacă** t != 0:

**atunci**: rezultat <-- rezultat + t

**sfârșit dacă**

inversează rezultatul

return rezultat

## Împărțirea unui număr cu o cifră

* *Date de intrare*: op1 (listă de cifre), op2 (cifră), baza
* *Date de ieșire*: câtul, restul (listă de cifre)
* *Precondiții*: -
* *Postcondiții*: -
* *Pseudocod*:

rezultat <-- []

t <-- 0

**pentru** i **de la** 0 **la** lungimea op1:

rezultat <-- rezultat + (t\*baza+op1[i])//op2

t <-- (t\*baza+op1[i])%op2

**sfârșit pentru**

**dacă** rezultat != [0]:

**atunci**:

ok <-- True

**cât timp** ok:

**dacă** rezultat[0] == 0:

**atunci**: șterge rezultat[0]

**altfel**: ok <-- False

**sfârșit dacă**

**sfârșit cât timp**

**sfârșit dacă**

return rezultat, t

## Conversia prin împărțiri succesive

* *Date de intrare*: număr (listă de cifre), baza inițială, baza finală (integer)
* *Date de ieșire*: număr nou (listă de cifre)
* *Precondiții*: -
* *Postcondiții*: -
* *Pseudocod*:

număr nou <-- []

**cât timp** număr != [0]:

rezultat <-- împărțire(număr,baza finală, baza inițială)

număr <-- rezultat[0]

număr nou <-- număr nou + rezultat[1]

**sfârșit cât timp**

inversează număr nou

return număr nou

## Conversia prin substituție

* *Date de intrare*: număr (listă de cifre), baza inițială, baza finală (integer)
* *Date de ieșire*: număr nou (listă de cifre)
* *Precondiții*: -
* *Postcondiții*: -
* *Pseudocod*:

număr nou <-- []

**pentru** i **de la** 0 **la** lungimea număr:

index <-- [număr[i]]

**pentru** j **de la** 1 **la** lungimea număr - i:

index <-- înmulțire(index,baza inițială,baza finală)

**sfârșit pentru**

număr nou <-- adunare(număr nou,index,baza finală)

**sfârșit pentru**

return număr nou

# Date de test

## Scenariu de rulare

|  |  |  |  |
| --- | --- | --- | --- |
|  | Utilizator | Program | Descriere |
| a |  | \*interfață\* | Interfață principală (conversii) |
| b | 1010101 |  | Alegerea unui număr |
| c | 2 |  | Selectarea bazei inițiale |
| d | 10 |  | Selectarea bazei finale |
| e | \*conversie\* |  | Apăsarea butonului pt conversie |
| f |  | 85 | Afișarea rezultatului conversiei |
| g | \*conversie cu baza intermediara\* |  | Apăsarea butonului pt conversie cu bază intermediară |
| h |  | 85 | Afișarea rezultatului conversiei |
| i | \*meniu\* |  | Alegerea nr instrucțiunii dorite |
| j | \*calcule\* |  | Terminarea programului |
| k |  | \*interfață\* | Interfață nouă (calcule) |
| l | 16 |  | Alegerea bazei |
| m | A16C |  | Introducerea primului număr |
| n | 7 |  | Introducerea celui de-al 2lea număr |
| o | x |  | Alegerea operației dorite |
| p | \*calculeaza\* |  | Apăsarea butonului pt calculul operației |
| q |  | 469F4 | Afișarea rezultatului calculului |
| r | \*X\* |  | Ieșire din program |

## Cazuri de testare

|  |  |  |  |
| --- | --- | --- | --- |
| Număr întreg | Din baza | În baza | Rezultat |
| 11111 | 2 | 10 | 31 |
| abc | 16 | 6 | 20420 |
| 1795 | 12 | 14 | 1077 |
| a11b2 | 3 | 9 | [număr invalid] |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Primul număr | Operație | Al 2lea număr | Baza | Rezultat |
| 1111111 | + | 101010 | 2 | 10101001 |
| a1b2c | - | ff | 16 | a1a2d |
| 786 | x | 4 | 9 | 3476 |
| 238597 | / | 8 | 14 | 3d42d r 1 |
| 33 | x | 3 | 2 | [număr invalid] |