# Taak 1A: Definieer die taak

## Onderwerp

Om n sakrekenaar te ontwerp en programeer wat basise berekeninge kan doen. Dit sluit plus, minus, maal en deel in. Dit gaan slegs een som op n slag kan doen.

## Deeglike beskrywing

Nie almal se hoofrekeninge is akuraat en vining genoeg nie. Dit kan lei tot vertraaging by die skool en werk. Dit kan ook lei tot tyds verlies. Kinders spandeer dus te veel tyd aan maklike somme soos plus, minus, maal en deel en dit veroorsaak hoe kognitiewe lading.

## Oplossing

Om hierdie problem op te los gaan ek n eenvoudige Delphi-sakrekenaar skryf. Ek skryf n eenvoudige sakrekenaar want die code sal dan vinniger ’’run’’ en minder spasie in ram opvat. Die sakrekenaar sal eenvoudig wees om te gebruik ; dus sal enige iemand weet hoe dit werk en dit sal tyd spaar. Die sakrekenaar sal se waarmee jy die meeste sukkel.

## Omvang van projek

Ek gaan die projek skryf in Delphi. Die sakrekenaar gaan daarop focus om mense te help om vinnige en akurate berekeninge te doen. Daar is verskeie probleme wat ek langs die pad gaan moet oplos. Bv gevalle waar gebruikers onvolledige of ongeldige toevoer aan die program deur gee bv. Deel deur 0. Ek beplan ook om n geheue stelsel te implimenteer saam met n stelsel wat vir gebrukers se waste somme hulle die meeste in die sakrekenaar ingevoer het.

# Taak 1B: Gebruikerstories

|  |  |  |
| --- | --- | --- |
| Rol(Wie) | Aktiwiteit(Wat) | Waarde(Hoekom) |
| Onderwyser | Wil andtwoorde kontroleer | Om die werk vining en akuraat te doen |
| Student | Vinnige wisk bewerkinge | Gaan help om vinniger en meer effektief te leer |
| Aktuaris | Maklik vinnige somme doen | Gaan vining eenvoudige probleme oplos |

# Taak 2: Aanvaardingstoets

Instruksie 1:

Die stelsel sal vir die gebruiker n ‘’ num-pad’’ voorsien waarop hulle, hulle som kan intik. Wanneer die gebruiker die som intik sal die stelsel die som ingeheel stoor as n string. Hierdie string sal elke keer opgedateer word wanneer die gebruiker n knopie druk.

Instruksie 2:

Die stelsel sal, n antwoord vir die som vertoon elke keer wat die leerder die = konppie druk. Dit sal ook die Antwoord berg om later vir die gebruiker te vertoon. Hierdie gebergde andw gaan ook gebruik word vir statestieke.

Instruksie 3:

Die stelsel sal invoer valideer deur te kyk of die string som leeg is. Die stelsel gaan okk seker maak dat daar ee teken(oporator) per som is.

Instruksie 4:

Die stelsel sal voor berei vir Error handling deur n lable. Aan die begin sal die lable leeg wees. As daar n error voorkom deur die validerings proses dan sal die ‘’vout’’ boodskap op die label vertoon.

# Taak 3: Navigasie tussen GGKs

Kyk heel agter.

# Taak 4A: Ontwerp die gebruikerskoppelvlak

Skerms screenshots

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

# Taak 4B: Datawoordeboek

Kort oorsig: Die program bied ’n eenvoudige sak rekenaar op **tsTwee** en wys geskiedenis (vorige berekeninge en foute) op **tsEen**. Die gebruiker tik die invoer som in; die uitdrukking word gebou in sVergelyking. Berekenings en foute word na tfSomme.txt geskryf.

**Hooflys (komponente en data-items)**

| **Naam** | **Tipe** | **Skop (Scope)** | **Doel / Opmerkings** |
| --- | --- | --- | --- |
| pgcMain | TPageControl | Komponent | Hou **tsEen** (geskiedenis) en **tsTwee** (rekenaar). |
| tsEen | TTabSheet | Komponent | Vertoon geskiedenis van somme in redVertoon. |
| tsTwee | TTabSheet | Komponent | Hoofrekenaar-koppelvlak (nommers, operatore, uitvoer). |
| btn0 … btn9 | TButton | Komponent | Voeg die syfer 0–9 by sVergelyking. |
| btnPlus | TButton | Komponent | Voeg ‘+’ by sVergelyking. |
| btnMinus | TButton | Komponent | Voeg ‘-’ by sVergelyking. |
| btnMaal | TButton | Komponent | Voeg ‘\*’ by sVergelyking. |
| btnDeel | TButton | Komponent | Voeg ‘/’ by sVergelyking. |
| btnMod | TButton | Komponent | Voeg ‘%’ (modulo) by sVergelyking. |
| btnAndw | TButton | Komponent | Voer die berekening uit; skryf resultaat/fout na UI en loglêer. |

## Plaaslike veranderlikes

**In btnAndwClick**

| **Naam** | **Tipe** | **Doel** |
| --- | --- | --- |
| cTeken | Char | Die enkele operator wat in die uitdrukking gevind is (+ - \* / %). |
| sGetal1, sGetal2 | string | Teksvorme van die twee operande links/regs van die operator. |
| iGetal1, iGetal2 | Integer | Omgeskakelde heelgetal-waardes van sGetal1 en sGetal2. |
| WaarIsOns | Boolean | “Skakelaar” om te weet of ons links (True) of regs (False) lees. |
| MeerAsEenTeken | Boolean | Valideer dat slegs **een** operator voorkom. |
| HeidigeKarakter | Char | Huidige karakter tydens parskring oor sVergelyking. |
| dAntwoord | Double | Eindresultaat van die berekening. |
| I | Integer | Bounded for-lus-indekser oor tekenreeks lengte. |
| iGetalGeRun | Integer | Teller van operators wat gevind is (verwagtend ≤ 1). |
| DelingDeurNul | Boolean | Vlag vir deling/modulo deur 0 (fouttoestand). |

**Lokaal vs Globaal (in hierdie projek)**

* **Globaal op vormvlak**: Alle visuele komponente (TButton, TLabel, TRichEdit, TPageControl, TTabSheet), asook die veranderlikes sVergelyking en tfSomme. Hierdie is direk beskikbaar vir enige procedure in TForm1.
* **Plaaslik**: Variables soos Delling deur nul is as plaaslik verklaar

# Taak 5: TVA en Validasie

**Toevoer (Input)**

| **Bron / Komponent** | **Beskrywing** | **Formaat** | **Datatipe** | **Validering van toevoer** | **Foutkontrolemeganismes** | **Waarheen gebruik?** |
| --- | --- | --- | --- | --- | --- | --- |
| btn0..btn9 | Voeg ’n syfer by die huidige uitdrukking sVergelyking | Karakter in [0–9] | Char -> bou string | Slegs syfers toelaat; maksimum lengte (bv. <= 20 karakters totaal) | Ignorer enige nie-syfer (kan UI blokkeer eerder as om by te voeg) | Bou operand(e) |
| btnPlus/Minus/Maal/Deel/Mod | Voeg **een** operateur by | Presies een uit + - \* / % | Char | Geen operateur as eerste of laaste karakter; | Indien tweede operateur gesien word: wys fout (“Kan net een som…”), kleur Label1 rooi, **log fout** | Merk operateur vir berekening |
| btnAndw (bereken) | Begin parse & eval | op1 op op2 sonder spasies | string na Integer/Double | Patroon: ^\d+[\+\-\\*\/%]\d+$; beide operande nie leeg nie; syfers binne Integer-reeks | **Deling deur 0**; TryStrToInt-misluk; ongeldige patroon; kleur/boodskap + **log fout** | Skryf resultaat na UI + log |
| Lêer tfSomme.txt (vir lees) | Laai geskiedenis | Reëls: <uitdrukking>=<resultaat of fout> | Teksreëls | Net leesbaar as lêer bestaan | try..except teen I/O-foute; leeg lêer -> wys niks | Vul redVertoon |
| btnNewT | Herlaai geskiedenis | – | – | – | Fout met Reset/ReadLn hanteer met | Vernuwing van geskiedenis |

**Algoritme 1: Ontleed & evalueer enkel-operateur-som**

**Doel:** Splits sVergelyking in x, operatuur, y, valideer, en bereken.

**Stappe:**

1. **Skandeer** sVergelyking links-na-regs; tel operateurs in {+,-,\*,/,%}.
   * As hoeveelheidop > 1: **FOUT**: “Kan net een som doen op ’n slag!”; kleur rooi; **log fout**; stop.
2. **Validasie**: geen operateur aan begin/einde
3. **Omskakeling**: gebruik StrToInt vir elk van op1, op2.
   * Indien enige faal: **FOUT**: “Ongeldige getal”; **log fout**; stop.
4. **Spesiale gevalle**:
   * As op='/' of '%' en op2=0 **FOUT**: “Kan nie deur 0 deel nie”; **log fout**; stop.
5. **Berekening**:
   * + op1 + op2 (Integer)
   * - op1 - op2 (Integer)
   * \* op1 \* op2 (Integer)
   * / op1 / op2 (Double, wys met FloatToStr)
   * % op1 mod op2 (Integer)
6. **Afvoer**: wys resultaat in Label1; **skryf log**: uitdrukking = antwoord; maak sVergelyking := ''.

**Algoritme 2: Geskiedenis skryf + lees + vertoon**

**Doel:** Hou ’n volhoubare log van alle pogings (resultaat **of** fout).

**Skryf (by berekening):**

1. Open tfSomme.txt met Append (of Rewrite as nie bestaan).
2. Skryf **een reël**: <sVergelyking>=<resultaat | 'Error: boodskap'>.
3. CloseFile op toepassing-afsluiting (of onmiddellik na skryf as jy wil “flush”).

**Lees (op btnNewT):**

1. redVertoon.Lines.Clear.
2. Reset(tfSomme) en **lees** reël-vir-reël (while not Eof do ReadLn).
3. Voeg elke reël by redVertoon.Lines.
4. Hanteer invoer/afvoer met try..finally.

**Opmerking:** Elkeen van bogenoemde loops het ’n **vaste boonste grens**: (kan nie vereweg aanhou nie)

* Skandeer loop <= Length(sVergelyking) karakters.

**TVA – Afvoer (Output)**

| **Afvoerplek** | **Vereistes** | **Formaat** | **Bron** | **Foute** |
| --- | --- | --- | --- | --- |
| **Label1** (TabSheet2) | Toon **huidige invoer** of **resultaat**; foutboodskappe in **rooi agtergrond** | Teks (Caption) | Realtime bou van sVergelyking; na berekening, eindresultaat | “Kan net een som…”, “Kan nie deur 0 deel nie”, “Ongeldige getal” |
| **tfSomme.txt** | Permanente rekord van alle berekeninge **en foute** | Uitdrukking+=+antw | Error: boodskap>` x | Skryf op elke btnAndw poging |
| **redVertoon** (TabSheet1) | Vertoon **hele geskiedenis** kronologies | Teks | Lees van tfSomme.txt | Leë of ontbrekende lêer -> wys net niks |

**Duidelike beskrywing van die toevoer**

* Die gebruiker kies syfers via **btn0..btn9** om ’n getal te bou.
* Die gebruiker kies **een** operateur via btnPlus/Minus/Maal/Deel/Mod.
* Die gebruiker bou die **tweede getal** via syferknoppies.
* Met **btnAndw** word die uitdrukking bevestig en bereken.

**Afvoervereistes vir twee hoofkoppelvlakke**

**1) Hoofskerm – tsTwee (Sakrekenaar)**

* **Label1** moet **invoer live wys** terwyl knoppies gedruk word (bv. “123+45”).
* Ná berekening moet **die resultaat** onmiddellik die hele Caption inneem (bv. “168”).
* **Foutstate**: Stel Label1.Color := clRed en wys ’n duidelike boodskap.
* **Knoppies** moet nie verdere karakters byvoeg nadat ’n fout gewys is\*\* totdat die gebruiker weer begin (kan skoonmaak deur nuwe invoer of ’n addisionele “Clear”-knop).

**2) Geskiedenis – tsEen**

* **redVertoon** moet **lees-alle** reëls uit tfSomme.txt en wys in volgorde van byvoeging.
* Formaat van elke reël is **konsekwent**:
  + **Sukses:** A+B=Resultaat (of ander operateur)
  + **Fout:** A/B=Error: Deling deur 0 ens.
* ’n “Vernuwe/Herlaai”-aksie (**btnNewT**) moet die boks **skoonmaak** en herlees.

**Valideringsreëls en Foutkontrole (samevatting)**

**Validering op toevoer**

1. **Patroon**: presies **een** operateur; geen operateur by begin/einde.
2. **Karakterstel**: slegs [0–9 + - \* / %].
3. **Lengte-limiet**: bv. maksimum 20 karakters vir sVergelyking.
4. **Datatipe**: x en y moet **ints wees**
5. **Deling/MOD**: tweede operand <> 0.

**Foutkontrolemeganismes**

* **Meer as een operateur** gee UI-fout + logreël “Error: Meer as een teken”.
* **Deling deur nul** gee UI-fout + “Error: Deling deur 0”.
* **Nie-numeriese operand / oorvloei** gee UI-fout + “Error: Ongeldige getal”.
* **Invoer/Afvoer-foute** op loglees/skryf gee vang met try..except en wys gebruikersvriendelike boodskap (bv. “Geskiedenis onbeskikbaar”).