Programozói dokumentáció

Numerikus integráló

Projekt felépítése:

A numerikus integráló projekt 9 darab fájlra támaszkodva működik:

1. numerical_integral.c
2. gui.c
3. gui.h
4. integral.c
5. integral.h
6. expression_parser.c
7. expression_parser.h
8. debugmalloc.h
9. CMakeListst.txt

A felsorolt fájlok közül az első a fő program, amelyben a menüvezérlés, a GUI futtatása és a numerikus integrálszámítás zajlik.

Felhasználói Felület:

A *gui.h* fájlban történik meg az interface megvalósításához szükséges könyvtárak hozzáadása a projekthez: gtk.h, string.h, stdio.h A GTK (GIMP Toolkit) külső könyvtár alkalmazásával készült az interface. Segítségével egy olyan, viszonylag kezdetleges, de a projekthez megfelelő, kalkulátorhoz hasonló kinézetű felület használható a program futása során, mely az integrálszámításhoz szükséges matematikai kifejezés és a kívánt intervallum megadását egyszerűsíti le, teszi átláthatóvá.

A header fájlban szerepel 4 darab struktúra definíciója: **Grids, Buttons, Entry, Labels,** melyek rendre a GUI nagy egységeit, gombjait, beviteli mezőit és a címkéit rendszerezik. Illetve itt található meg a szükséges függvények prototípusai is.

A külső könyvtár alapvető adatszerkezete a **GtkWidget**, amely egy grafikus felhasználói felületelemet képvisel. Ez a struktúrája minden widgetnek a GTK-ban.

A **gpointer** egy általános pointer típus ebben a könyvtárban, amely egy nem meghatározott típusú adatra mutató pointerre alkalmazható. Gyakran használatos a GTK eseménykezelésében az eseményekhez tartozó adatok továbbítására a visszahívó (G_CALLBACK) függvényeknek.

A *gui.c* a felület megvalósításáért felelős függvényeket tartalmazza:

- **void insert_text(GtkWidget *button, gpointer user_data)**
  - nincs visszatérési értéke
  - paraméterei egy GtkWidget típusú gombra mutató pointer és egy gpointer
  - adott gombra való kattintás után a megfelelő karaktert vagy fordított lengyel jelölést illeszti be a beviteli mezőbe

Programozói dokumentáció

- **void save_to_file(GtkWidget *button, gpointer user_data)**
  - nincs visszatérési értéke
  - paraméterei egy GtkWidget típusú gombra mutató pointer és egy gpointer
  - kiírja a functions.txt fájlba az első beviteli mezőben található szöveget az OK gomb hatására
- **void save_interval(GtkWidget *button, gpointer user_data)**
  - nincs visszatérési értéke
  - paraméterei egy GtkWidget típusú gombra mutató pointer és egy gpointer
  - kiírja a functions.txt fájlba a megadott intervallumot
- **void disable_button(GtkWidget *button, gpointer user_data)**
  - nincs visszatérési értéke
  - paraméterei egy GtkWidget típusú gombra mutató pointer és egy gpointer
  - egy adott gombot tilt le
- **void over(GtkWidget *button, gpointer user_data)**
  - nincs visszatérési értéke
  - paraméterei egy GtkWidget típusú gombra mutató pointer és egy gpointer
  - adott gomb hatására bezárja a GUI-t

Az *integral.h* fájlban történik a következő könyvtárak include hívása: string.h, math.h, stdio.h, stdlib.h, ctype.h, stdbool.h. Definiálva van egy INITIAL_SIZE szám 256 értékkel, amely a fájl olvasásakor egy kezdetleges méretet szab meg, hiszen előre nem tudjuk a beolvasandó fájl méretét, így ettől nagyobb érték esetén a definiált számot megkétszerezzük minden alkalommal. Ez memóriafoglalás szempontjából is hasznosítható. Szinten szerepelnek itt is megfelelő függvénydefiníciók.

Az *integral.c* függvényei a főprogrambeli integrálszámítást készítik elő:

- **void read_file(const char *filename, char **last, char **second_last)**
  - nincs visszatérési értéke
  - paraméterei egy string, és két stringre mutató pointer
  - a függvény az első paraméterként kapott stringben álló fájl utolsó két sorát olvassa be, majd adja vissza cím szerint azokat
- **void remove_spaces(char* str)**
  - visszatérési értéke nincs
  - paramétere egy string
  - a string kezdetén és végén található space-eket levágja
- **double find_supremum(Node* expr, double start, double end, double step)**
  - visszatérési értéke double típusú
  - paraméterei egy bináris fa, és három valós szám
  - az első szám az intervallum kezdőpontja, a második a végpont, a harmadik pedig a lépték
  - a függvény a fában kapott matematikai kifejezésnek az adott intervallumon meghatározza a szuprémumát és ezt is adja vissza

Programozói dokumentáció

Az *expression_parser.h* fájlban vannak definiálva azon struktúrák, amikre alapszik a fordított lengyel jelölésű matematikai kifejezések kiértékelése. A változók, operátorok, számok és függvények önálló struktúrákkal bírnak, melyek egy állapotgép és egy stack (verem) segíségével egy bináris kifejezésfát építenek fel. A megfelelő függvények definíciója is itt található.

Az *expression_parser.c* fájlban a következő függvények szerepelnek:

- **void push (NodeStack* stack, Node* node)**
  - a stack tetejére helyezi a kifejezés fa egy elemét
  - ha a verem tele van, hibát ad
- **Node* pop(NodeStack* stack)**
  - egy fa elemet eltávolít a stack tetejéről és értékét visszaadja
  - ha üres a stack, hibát ad
- **Node* create_variable(char c)**
  - megfelelő pointerekkel a megfelelő helyre épít a fába egy változót
  - hasonlóan tesznek a következő függvények:
- **Node* create_number(int n)**
- **Node* create_function(const char* name, Func func)**
- **Node* create_operator(char operator)**
- **Node* parse(char* expression)**
  - egy fordított lengyel jelölésű matematikai kifejezésből épít fel egy fát
- **double evaluate(Node* head, double x)**
  - egy fában kapott matematikai kifejezést értékel ki az x változó helyén
- **void free_tree(Node* node)**
  - felszabadítja a paraméterként kapott dinamikusan foglalt bináris fát

A szabványos könyvtárakon kívül csupán a GTK külső könyvtár megléte szükséges a program fordításához. A projekt build rendszerét a CMake segítségével konfiguráljuk. A CMake egy platformfüggetlen építési rendszer, amely egyszerűsíti a projektek konfigurálását és építését. A *CMakeLists.txt* fájlban található konfigurációs parancsok határozzák meg a projekt szerkezetét, a forrásfájlok helyét és az építési paramétereket.