Koordinátageometria - Programozói Dokumentácó

A program célja:

Egy olyan grafikus program, amelyben a felhasználó által megadott alakzatokat egy koordinátarendszerben megjeleníti, és ezekkel egyszerű szerkesztéseket lehet végezni.

Három alakzat közül lehet választani: szakasz, egyenes, kör. Szerkesztések: metszéspontok elhelyezése, merőleges állítása.

A koordinátarendszer négyzet alakú, közepe az origó, és a [-25,+25] intervallum látszódik az x, y koordinátán.

Lehetőség van rajzot betölteni és elmenteni.

Nyelv: C (C11)

Grafikus megjelenítő: SDL2

Operációs rendszer: Windows, Linux, MacOs

Memóriaigény: 8 Mb

Progra felépítése, modulok:

main.c:

Itt van a main függvény, plusz pár segédfüggvény. A main meghívja az ablak és a gombok felállítását. Segédvátlozókban tárolok állapotokat (gépelhet, quit, vár_kattintást), lenyomott gomb indexét és beviteli szöveget. Létrehozok az alakzatoknak és a metszéspontoknak egy pointert, láncolt lista eleje lesz (külön struktúra van a 2 típusra). Esemény ciklus: esememény alapján állapotok változtatása (ha kell), vezérlőben levő függvények meghívása. Ciklusból kilépve listák felszabadítása, sdl bezárása.

megjelenites.c:

Grafikus felület felállítása. Gombok, alakzatok, metszéspontok megjelenítése.

vezerlo.c:

Láncolt listák kezelése. Kattintás-ra meghívja a kellő függvényt (eger_lenyomva). Gombok és szövegdobozok módosítása. Gépelés kezelése.

szerkesztesek.c:

Metszéspontok kijelzése és merőleges állítása.

filekezeles.h:

Mentés és betöltés.

Adatstruktúrák:

Ablak: window (az ablak megjelenítéséhez kell), renderer (megjelenítő, ide lehet rajzolni), ablak mérete (ábra méret a négyzet alakú koordinátarendszer mérete pixelben, menü szélesség).

Gomb: méret (sdl négyzetként tárolom), szoveg (mi legyen a gombon írva (változhat futás közben), en (engedélyezés - meg kell e jeleníteni).

Alakzat (láncolt lista): títpus (semilyen, szakasz, egyenes, kör), két pont koordinátája - egész (szakasznál muszály két pont, egyenesnél és körnél csak az elsőt használom (x1, y1)), r - sugár és szög (körhöz és egyeneshez (meredekség) kell), következő elem.

Metszespont (láncolt lista): pont koordinátái - valós, következő elem.

Szerkesztések:

Merőleges: Szakaszra és egyenesre lehet merőlegest állítani. Egyenes esetén összehasonlítja az egyenes meredekségét annak a szakasznak a meredekségével, amint az egyenes egyik pontjából a kattintás helyére húzok. Ha elég kicsi ez a meredekség különbség akkor a kattintott pontban állítok egy egyenest, aminek 90 fokkal nagyob a meredeksége az egyenesnél. Szakasznál kiszámolom a szakasz meredekségét, és azt vetem össze a kattintott pont és a szakasz egyik végét összekötő szekesz meredekségével. Meg vizsgálom itt még azt is, hogy a kattintott pont bele esik e a szakasz értelmezési tartományába és érték készletébe.

Metszéspontok: Minden lehetséges alakzatpárra meghívom a hozzá tartozó függvényt (kör-kör, egyenes/szakasz - kör, egyenes/szakasz - egyenes/szakasz). Egyenesből szakasz csinálok minden esetben (a két összekötő pont az ábra két szélén lesz, nem marad le semmi). A metszéspontok kiszámolásához használt képletek forrásai:

Kör - kör: http://2000clicks.com/MathHelp/GeometryConicSectionCircleIntersection.aspx

Kör - szakasz: http://csharphelper.com/blog/2014/09/determine-where-a-line-intersects-a-circle-in-c/

Szakasz – szakasz: https://en.wikipedia.org/wiki/Line%E2%80%93line_intersection

Függvények:

main.c:

void gombok_adatai(Gomb *gombok) - Gombok adatainak beállításavoid melyik_alakzat(Alakzat_tipus *tipus, int lenyomott_gomb) - Alenyomott gomb alapján visszatér, hogy melyik alakzat gombja az.

bool alakzat_gomb(int lenyomott_gomb) - Alakzat-e a lenyomott gomb (kör, szakasz vagy egyenes gomb lett lenyomva).

megjelenites.h:

double szog_radianba(int szog) - Átváltja a megadott szöget fokból radiánba.

void init() - SDL inicializálása.

void pont_koordinata_atvaltas(Ablak *ablak, double *x, double *y) - Kapott pont koordinátáit átváltja az ablak koordinátáiba.

SDL_Window *window_letrehoz(int szel, int hossz, char *fejlec) - Ablak felállítása méret és fejléc alapján.

SDL_Renderer *renderer_letrehoz(SDL_Window *window) - Megjelenítő létrehozása.

void hatter_beallit(Ablak *ablak) - Háttérszín fehérre állítása.

void kezdo_vonalak_rajzolas(Ablak *ablak) - 4 vonal meghúzása: 2 egymáshoz közeli ami elválasztja az ábrát a menütől, a másik 2 pedig az x és y tengely.

void szoveg_kiir(Ablak *ablak, Gomb *gomb) - A kapott gombon megjeleníti a hozzá tartozó szöveget.

void gomb_megjelenit(Ablak *ablak, Gomb *gomb, bool lenyomva) - Gomb megjelenítése, meghívódik lenyomásnál és felengedésnél, hogy a szín változtatásával legyen egy kis animáció.

void szakasz_rajzolas(Ablak *ablak, Alakzat *szakasz) - A kapott két pontot átválja az ablak koordinátáiba, mejd összeköti őket.

void egyenes_rajzolas(Ablak *ablak, Alakzat *egyenes) - A megadott pont és a szög alapján létrehoz két pontot az ábra jobb és bal szélén, majd szakaszként rajzolja/tárolja az egyenest.

void kor_rajzolas(Ablak *ablak, Alakzat *kor) - Kör kirajzolása. Sugár távolságban helyez el pontokat a középpont körül .

void metszespont_rajzolas(Ablak *ablak, double x, double y) - Metszéspont kirajzolása, a kapott x y koordináták alapján.

void ablak_felallit(Ablak *ablak) - Ablak adatainak beállítása, megjelenítés.

vezerlo.c:

void alakzat_listaba(Alakzat_tipus tipus, int x1, int y1, int x2, int y2, int r, int szog, Alakzat **eleje) - A kapott adatok alapján létrehoz egy új alakzatot, és elmenti azt a lista elejére.

bool metszespont_elmentve(Metszespont *eleje, double x, double y) - El lett-e mát mentve a metszéspont a listába

void metszespont_mentes(Metszespont **eleje, double x, double y) - Elment egy új metszéspontot a láncolt lista elejére.

void metszespont_felszabadit(Metszespont **eleje) - Felszabadítja a metszéspontokat tároló láncolt listát.

void metszespontok_megkeresese(Ablak *ablak, Alakzat *eleje, Metszespont **pontok) - Összehasonlítja az összes lehetséges alakzat párt, hogy van e metszéspont.

void eger_lenyomva(Ablak *ablak, Gomb *gombok,int x, int y, int *lenyomott_gomb, Alakzat **eleje, Metszespont **pontok, bool *varakozas_kattintasra) - Kattintás kezelése, megkeresi a lenyomott gombot, meghívja a gombokhoz tartozó funkciókat.

SDL_Event varakozas_esemenyre() - Várakozás, majd visszatérés az esemény-el.

int string_to_int(char *szoveg) - szöveget egésszé vált, a szövegben csak számok lehetnek, és egy negatív jel a legelején.

void gomb_adat(Gomb *gomb, int x, int y, int w, int h, char *szoveg, bool en) - Gomb struktúrába betölti a megadott adatokat.

int melyik_gomb(Gomb *gombok, int x, int y) - A megadott x y koordinátákból megállapítja, hogy melyik gombon van az (x, y) pont.

void gombok_8_13_engedelyezes_megjelenites(Ablak *ablak, Gomb *gombok, int melyiket_ne) - Az alazak gombok alatti 4 gomb engedélyezése, és kirajzolása, kivéve azt az 1-et, ami nem kell. Ha mind a 4 kell, akkor olyan indexűt adok meg, ami nem >= 8 & < 13.

void negy_szovegdoboz(Gomb *dobozok, char *a, char *b, char *c, char *d) - Négy szövegdoboz beállítása, megadott sztringek belepakolása.

void szakasz_elokeszit(Ablak *ablak, Gomb *gomb) - Egy szakasz megadásához szükséges gombok, szövegek felállítása.

void egyenes_elokeszit(Ablak *ablak, Gomb *gomb) - Egy egyenes megadásához szükséges gombok, szövegek felállítása.

void kor_elokeszit(Ablak *ablak, Gomb *gomb) - Egy kör megadásához szükséges gombok, szövegek felállítása.

void elokeszites_torles(Ablak *ablak, Gomb *gombok) - Az alakzatok beviteléhez szükséges gombok és szövegek törlése.

bool kell_input(int lenyomott_gomb) - A megkapott gomb index-e alapján megállípítja, hogy van e ott input lehetőség. A 0. gomb a filnév, a 8-11 gombok az alakzatok adatának beviteli helye. Tehát az ezeken kívüli gombokhoz nem kell input.

void gepeles(Ablak *ablak, Gomb *gomb, char *bevitel) - Hozzáadja a gomb szövegéhez a lenyomott karaktert, és újra kirajzolja a gombot.

void karakter_torles(char *szoveg) - Kirötli az utolsó kerektert a kapott stringből.

bool kell_e_rajzolni(Alakzat_tipus alakzat) - Megmodja, hogy a megadott alakzat típus szerint kirajzolható.

void alakzat_rajzolas(Ablak *ablak, Gomb *gombok, Alakzat_tipus alakzat, Alakzat **eleje) - A kapott alakzatot elmenti a listába, majt meghívja, a kirajzoló függvényt hozzá.

void lista_felszabadit(Alakzat **eleje) - Az alakzat lista felszabadítása.

void kiterjesztes_torlese(char *szoveg) - kiröli a ".txt"-t a filenév végéről, miután már betöltöttük a file-t.

szerkesztesek.c:

int meroleges_szoge(int x1, int y1, int x2, int y2) - Két pontból megállapítja az összekötő szakaszra merőleges egyenes meredekségét.

void meroleges_allitas(Ablak *ablak, Alakzat *alakzat, int x, int y) - A kapott pont és alakzat alapján meghívja a szakasz rajzolást, úgy hogy az merőleges legyen.

void alakzat_keres(Ablak *ablak, Alakzat *eleje, int x, int y) - Megkeresi melyik alakzatra kattintottunk, majd abban a pontban merőlegest állít az egyenesre/szakaszra.

void metszespont_kor_kor(Ablak *ablak, Alakzat *kor1, Alakzat *kor2, Metszespont **pontok) - Két kör metszéspontját megkeresi . Képlet forrás: http://2000clicks.com/MathHelp/GeometryConicSectionCircleIntersection.aspx

bool szakasz_eleme(Alakzat *szakasz, double x, double y) - Megmondja, hogy benne van e az adott pont a szakasz értelmezési tartományában és értékkészletében.

void egyenesbol_ket_pont(Alakzat *egyenes, double *p1x, double *p1y, double *p2x, double *p2y) - Egyenesből felvesz 2 pontot szögfüggvények segítségével az ábra két szélén.

void metszespont_kor_egyenes(Ablak *ablak, Alakzat *kor, Alakzat *egyenes, Metszespont **pontok) - Egy kör és egy szakasz/egyenes metszéspontját megkeresi. Egyenlet forrás: http://csharphelper.com/blog/2014/09/determine-where-a-line-intersects-a-circle-in-c/

void metszespont_egyenes_egyenes(Ablak *ablak, Alakzat *e1, Alakzat *e2, Metszespont **pontok) - Kiszámolja két egyenes metszéspontját. Képlet forrás: https://en.wikipedia.org/wiki/Line%E2%80%93line_intersection

void metszespont_kor_kor_ellenorzes_javitas(Alakzat *kor1, Alakzat *kor2, double *x1, double *y1, double *x2, double *y2) - Leellenőrzi a két metszéspontot, hogy rajta van e a két körön, ha nincs, akkor végigmegy a egyik körvonalon, és megnézi hol van sugár távolságban a másik körtől (átállítja a metszéspontokat)

filekezeles.c:

void mentes(char *file_nev, Alakzat *eleje, Metszespont *pontok) - Elmenti a megadott file-ba az alakzatokat és a metszéspontokat.

void betoltes(Ablak *ablak, char *file_nev, Alakzat **eleje, Metszespont **pontok) - Betölti a megadott file-ból az alakzatokat és a metszéspontokat.

Program futtatása:

Windows: CodeBlock és SDL2 telepítésével, új sdl2 projektbe behelyezve a kód file-okat lehet lefordítani.

Linux: GCC (sudo apt-get install gcc) és SDL2 (sudo apt install libsdl2-dev libsdl2-gfx-dev libsdl2-image-dev libsdl2-ttf-dev libsdl2-mixer-dev) telepítése után ezt a parancsot kell futtatni:

```
gcc main.c megjelenites.c vezerlo.c szerkesztesek.c filekezeles.c -o Koordinatageometria -lm `sdl2-config --cflags --libs` -lSDL2_gfx -lSDL2_ttf -lSDL2_image -lSDL2_mixer
```

Megjegyzés:

- Ha a metszésontokat nem pontosan jeleníti meg, akkor el kell menteni, és betölteni (nem kell közte bezárni)

Bug fixek (2018.11.24):

- Lehet függőleges egyenest állítani: megvizsgálja, hogy a megadott szöggel függőleges lenn. Ha igen akkor az összekötött pontok x koordinátája nem változik a megadottól, az y pedig az ábra tetején és alján lesz (megjelenites.c - egyenes_rajzolas() függvény)
- Merőleges állítása egyenesre: a szögvizsgálat minden esetben működik most már (szerksztesek.c - alakzat_keres())
- Metszéspontok többszörös mentése: most már csak akkor fogja elmenteni, ha nem találta meg a metszéspont listában (vezerlo.c - metszespot_mentes() és metszespont_elmentve())
- Egyenes problémák: nem sikerült az összes egyenes típussal fellépő hibát javítani, ezért szakaszként tárálom az egyeneseket (megjelenites.c - egyenes_rajzolas(), átváltom szakaszba, és a szakasz_rajzolas()-al jelenítem meg, mert így pontos a rajz)

Készítette - Nagy Roland

SYQW1F