PacMan programmer manual

The game contains 3 C and 2 header files.

## Main.c:

int main(int argc, char \*argv[])

{

Inicialization(window,sound,timer, music, icon) Giving values to surfaces and checking if they could open. Place of GameState enum, which will tell us if we’re going to enter the menu, exit, or join the game (depending ont he returning value). Also contains the liberations.

}

void nyertel(void) //win

{

If we won the game, this function get called. Zooming the „youwin” picture (origo centrally) in 50 steps. 20ms delay between each step. Also dimming the background

}

void vesztettel(void) //lose

{

Same as Win, but get called if we lose.

}

int highscore(void)

{

After the win/lose function, we can enter our name. The game accepts characters between a-z, A-Z, and 0-9. It uses dinamic memory allocation. At the event of pushing a button, the previous and the current characters are printed out. If you hit backspace the current name is being deleted.

}

void highscorefajlba(char\*nev) //Highscore to file

{

Loading the current highscore file into a linked list. Inserting our current result into it’s space (depending ont he point) and writes the new file.

}

## Menu.c:

int menu\_kezdo(void)

{

Inicializálja a menüt, kirajzoltatja a háttérképet, az ikonokat, szövegeket, majd eventre vár, és a beérkező eventet szétválogatja. Kattintásra az egér koordinátái szerint dönt, melyik függvényt kell meghívja. Új játék és játék betöltés esetén beolvassa a fájlból a térképet.

}

void menu\_hatter\_kirajzol(void)

{

Kirajzol egy fekete képet, majd a hátteret, a címet, és a hang ikont.

}

int menu\_leiras(void)

{

Kiírja a játék leírását tartalmazo tömb tartalmát, illetve a pontoknál a képeket is. Mint minden menüpontban ebben is van eventre várás, melynek céla, hogy a hang változtatható legyen.

}

int menu\_eredmenyek(void)

{

Beolvassa fájlból a highscore-t és kiíratja azt a képernyőre. Szintén van event várás.

}

## Jatek.c:

int jatek\_indit(void)

{

a játék indító függvénye, megkeresi Blinky és Pacman helyzetét, ebből kiszámolja koordinátájukat. Amennyiben előzőleg menü volt, vissza teszi Blinky alatti eredeti értéket. Kezeli az időzítőket, és eventet vár. A játékban az event várásban benne van, hogy a hangot is át tudja állítani. Ha PeekEvent van, Blinky következő irányát számoltatja ki, illetve extra pontokat generáltat(lásd specifikáció) Userevent esetén törli pacman heylzetét, kiszámolja aktuális pálya[][] pozícióját (blinkynek is) majd megvizsgáltatja, hogy nem esik-e pálya szélébe vagy falba a dolog. Irányok szerint új koordot is számíttat. Megvizsgálja nyertünk/vesztettünk-e. bemenet szerint kezeli a cheatet, irányítást illetve a konami-t. Eventet vár.

}

int jatek\_menu(void)

{

ESC lenyomására jelenik meg, 4 menüponttal. Szokásos némító/halkító/kilépő evenetre vár. Kattintás esetén egér koord. alapján dönt milyen esemény fusson le. Kilépésnél meghívja a kilepes fv.-t mely ellenőrzi, mentettünk-e.

}

void kirajzol(void)

{

Végig pörgeti a pályát, és minden karakternek megfelelő képet rajzol az adott helyre( #fal, 1PacDot, 2PowerDot, 3Cherry,4Orange, 5Melon, 6Bell, 7Key)

}

int kilepes(void)

{

Megkérdezi, hogy biztos mentés nélkül akarunk-e kilépni, és ha nem akkor vissza dob minket a játékmenübe.

}

int almenu\_start(void

{

Módosított GameState enum, mellyel az almenü felstackelődésést lehet kiküszöbölni.

}

void palyaszele(void)

{

Megvizsgálja, hogy PacMan ill. a szellem pozíciója elérte-e a pálya elvi széleit(azért elvi, mert ha fal van akkor már nem tud kimenni addig) és ha igen, és az irány ugyan arra mutat, akkor áthelyezi az illetőt a pálya túloldalára.

}

void falvizsgalo(void)

{

Amennyiben a következő irány nem esik falra, átállítja a jelenlegi irányenumot illetve a sebességvektorokat. Amennyiben a jelenlegi irány következő lépése falra mutat( és a koordináta is egységre esik) megállítja pacmant.

}

void pontrendszer(int y,int x)

{

PacMan éppen elfoglalandó pozíciójának nézi meg a tartalmát. Amennyiben az pontot ér, növeli azt, és ki rendereli a pontszámot.

}

void kovkoordinataszamolo(void)

{

Kiszámolja a következő koordinátákat.( jelenlegi+sebességvektor)

}

void kiment(void)

{

Kimenti az egész pályát és a pontot egy save.sav fájlba. A fájlban a tartalom szövegként tárolódik( akár meg is tekinthető tcmd-vel)

}

void extrapontok(void)

{

A specifikációnak megfelelő esélyekkel generál egy extra pontot érő számot, és ezt beilleszti a pályába. a beillesztés csak üres vagy sima pont helyére történhet, illetve ha 10 próbálkozás után sem talál ilyen helyet, akkor a játék folyamatosságának megőrzése érdekében elvetésre kerül az aktuális extra pont.

}

void blinkykoord(void

{

Megegyezik a falvizsgáló fv.-nyel, csak ez Blinky irányát és sebességét változtatja.

}

Megjegyzés: Amennyiben a program írásának közel egy hónappal ezelőtti pillanatában lett volna fogalmam a generikus fv. hívás létezéséről, pár sorból be tudtam volna tenni mind a négy szellemet. Azért van csak egy szellem, mert nincs arcom mind a négyet csak nevében és pár betűben más fv.-ek favágós ctrl-c+ctrl-v módszerével betenni, a generikus átírásra pedig nem volt elég időm.