|  |  |
| --- | --- |
| Картина, която съдържа текст, вентилатор, устройство  Описанието е генерирано автоматично | **ВИСШЕ ВОЕННОМОРСКО УЧИЛИЩЕ „Н. Й. ВАПЦАРОВ“**  ***9002 Варна, ул. „В. Друмев“ 73, тел.052/632-015, факс 052/303-163*** |
| ***“FILII MARIS SUMUS”***  **ФАКУЛТЕТ „ИНЖЕНЕРЕН“**  **КАТЕДРА „ИНФОРМАЦИОННИ ТЕХНОЛОГИИ или ЕЛЕКТРОНИКА“** |
| A picture containing text  Description automatically generated |  |

**KУРСОВ ПРОЕКТ**

**по дисциплината „Основи на програмирането и алгоритмизация“**

**теМА: „CLASSIC ARCADE GAME CLONE: RECREATING A CLASSIC ARCADE GAME LIKE PAC-MAN OR TETRIS“**

**Студент: Дамян Веселинов Чакъров**

**Специалност: Софтуерни и Мрежови технологии**

**фак. № 141-24115**

**гр. Варна**

**2024 г.**

Този проект е терминално пресъздаване на аркадната игра Тетрис на езика C. Тетрис представлява поле, което се изпълва с фигури, подредени от играча, които са винаги с абсолютен размер 4 блокчета и се различават единствено по подредбата и цвета им.

**Техническа документация**

Програмата е съставена от 3 файла:

* main.c
* functions.c – изпълнението на всяка една от функциите в програмата, като двете основни, към които се директно се достъпва от main файла, са play() и scoreboard().
* globals.c – декларирането на всяка глобалните променливи за програмата се случва тук, както и дефинирането на константи.

Допълнителни 2 header файла:

* functions.h – инициализацията на функциите, описани в едноименния .c файл.
* globals.h – служи като връзка между globals.c и останалите файлове в програмата, за да може да се достъпва в променливите.

В програмата освен двата създадени от мен header файлове са използвани и допълнително 5, от които:

<stdio.h>

Използвани са стандартните функции за запис и четене от конзолата/файла.

<stdlib.h>

Използвани са scanf(), system(), rand() и srand().

<stdbool.h>

Използвани са променливите от типа bool.

<string.h>

Използвана е функцията memcpy().

<unistd.h>

Използвана е функцията sleep().

<time.h>

Използвана е time().

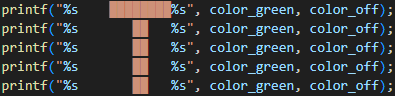
<conio.h>

Използвани са getch() и kbhit()

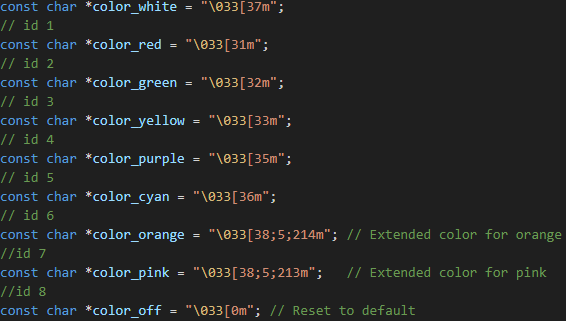
**Изпълнението на main() в main.c файла:**

Започва с декларирането на char input;, в която ще се съхранява входа от потребителя. След което е използвана функцията system("cls");, за да се изчистят предходни команди и входове в терминала.

Редовете 22-26 изписват заглавното име на играта TETRIS в различни цветове, за пример с Т:

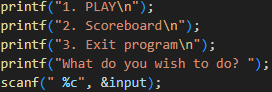


символът на издължен правоъгълник, който е използван е ограден от две %s, чрез които се вмъкват константните променливи от globals.c (с помощта на globals.h), съотвестващи на избрания цвят (или липса на такъв и отново върнати към стандарта).

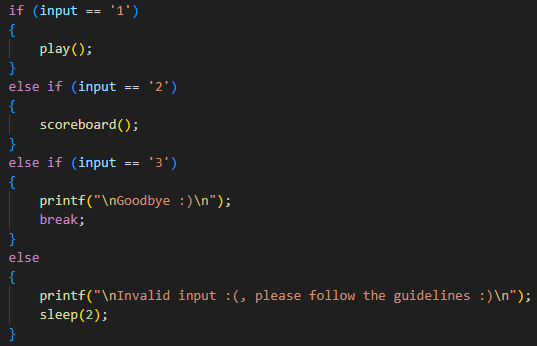


Декларацията на константите за цвета следва ANSI принципа, че първата част от декларацията при всеки един от цветовете \033[ инструктира терминала за предстояща промяна на цвета, стила или позицията, а следходното е желаната промяна, в случая са кодовете за всеки от съответните цветове.

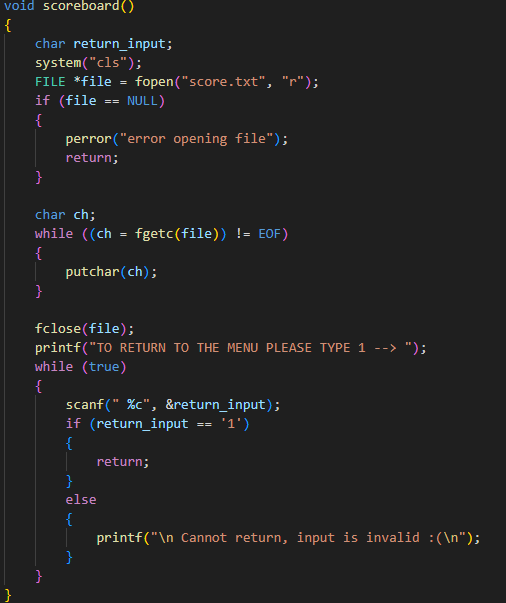
След изписването на заглавното лого на приложението следват

опциите, които са представени на потребителя, като след това се разчита избраното и се записва в адреса на променливата. Функцията разчита всичко въведено от потребителя, включително и новия ред Enter, за да се избегнат нежелани входове в променливата е поставено пространство преди %c, което сигнализира на функцията да пропусне whitespace characters.

Проверката на входа се изпълнява по следния начин:

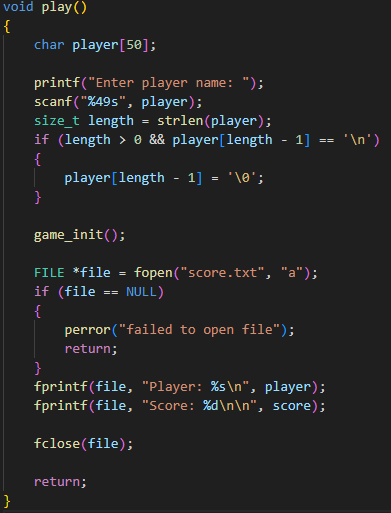


При начален вход ‘3‘, тоест изход, се излиза ефективно от програмата и се завършва успешно изпълнението.

При начален вход ‘2‘, се извиква функцията scoreboard() от functions.h като същинския код на функцията се съдържа в functions.c.

В нея - декларира се променлива char return\_input;, след което се изчиства терминала и се отваря файла score.txt за прочит. Прави се проверка дали е успешно отварянето на файла, ако не – return;, а в противен случай се изписва цялостния файл чрез обхождане. Изисква се вход ‘1‘ за да се върне програмата обратно в менюто.

При начален вход ‘1‘ се извиква функцията play(), от която следва същинската част на проекта.



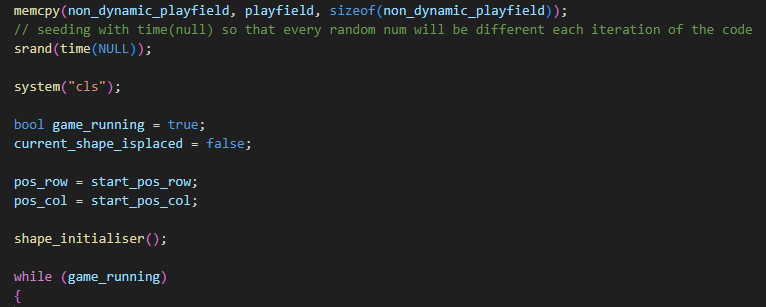
Тя започва с деклариране на масив от типа char, където ще се съхрани името на потребителя, като чрез if() проверката се премахва новия ред.

След успешното съхранение на името се извиква функцията game\_init(), която е входната точка на самата игра.

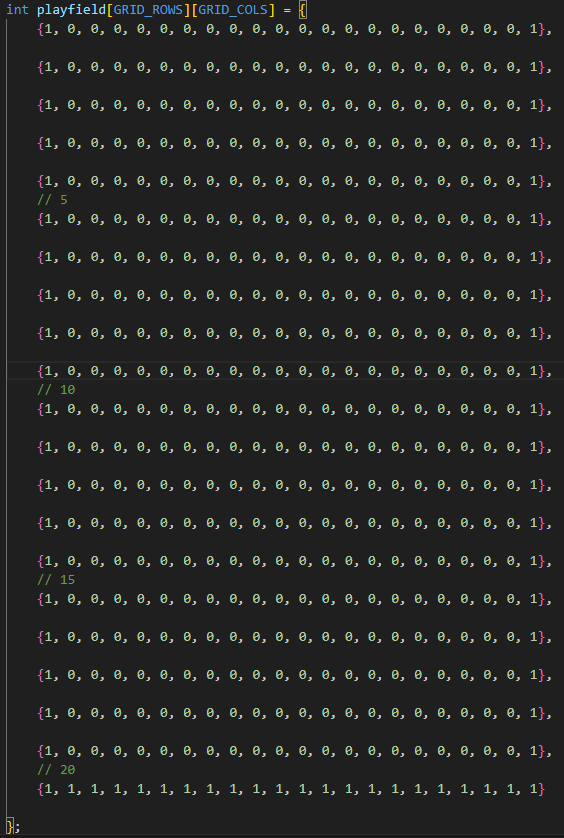
След прекратяването на играта, се отваря файлът score.txt, но тук със свойството append (да се изписва текст на края на файла). Прави се проверка дали успешно е отворен файлът – ако да - се изписвсат името на играча и неговия резултат, използвайки глобалната променлива score , след което файлът се затваря и потребителят е отново върнат в менюто на програмата.

game\_init()

Това е функцията, в която върви самият цикъл на играта.



Преди влизането в самия while(), се прави подготовката за първата инстанция в играта :



Копират се стойностите на двумерния масив playfield[][] в преди това празния non\_dynamic\_playfield[GRID\_ROWS][GRID\_COLS];, където 0 обозначава празните позиции, а с 1 самите граници на полето за игра. Константите по които са дефинирани двата двумерни масива

#define GRID\_ROWS 21

#define GRID\_COLS 22

го определят по такъв начин, че ако се премахнат двете колони с единици за самата клетка и последния ред, би се образувал квадрат от нули. Тъй като символът който се използва за изобразяването на всяка една от фигурите от полето :

█

е по-висока отколкото широка, за да се постигне желаната формана кубче, поставяме два пъти този символ

██

, което е и причината полето, което на пръв поглед е правоъгълно, реално да е квадрат.

Булевата променлива, която контролира дали играта ще върви се инициализира и задава на true, освен това задаваме false на проверката за текущата фигура.

Координатите на които ще се призове тази именно фигура се задават на стартовите и се изви