Задача: игра с приключенец

Курсова задача 2, Училищен курс по програмиране на C# 2013/2014

# Описание

Да се разработи конзола игра в реално време, в която играчът е приключенец в подземие и целта му е да събере максимално количество точки за ограничено време и да продължи към следващото ниво. Нивата са случайно генерирани и съдържат изход към следващото ниво, различни препятствия, както и различни контейнери за преравяне, в които може да има различни бонуси или капани за играча. Играчът започва играта с определено количество точки живот и има фиксирано време за решаването на всяко ниво. Играта приключва в два случая: когато точките живот на играча намалеят до нула или когато времето за решаване на нивото изтече, преди играчът да е преминал през изхода към следващото ниво. След приключване на играта, на играча се предоставя информация за общия брой точки, които е събрал.

# Изисквания

## Функционални изисквания

* Игралното поле представлява правоъгълна мрежа от символи;
* Външният ръб на всяко ниво винаги се състои от стени, така че играчът да не може да напусне пределите му;
* На случайни места в полето са поставени стени, колони и други препятствия с варираща дължина и ширина;
* На случайни места в полето са разположени различни контейнери, които могат да се преравят; ето два задължителни примера:
	+ Купчини с боклуци;
	+ Сандъци (могат да бъдат отключени или заключени);
* Поведението на разлините контейнери при преравяне трябва да се различава, например купчините с боклуци може да отнемат 2 секунди за преравяне, сандъците – 1, но само в случай, че са отключени; за заключените сандъци потребителят трябва да може да направи избор дали да ги отключи с шперц, което би отнело 3 секунди, в допълнение към секундата за преравяне на вече отключения сандък;
* Вътре в контейнерите, които могат да бъдат преравяни, се съдържат по един или повече предмета, които могат да имат положителен или отрицателен ефект върху играча; ето няколко задължителни примера:
	+ Различни видове скъпоценности, които дават определен брой точки на потребителя;
	+ Бомби, които автоматично избухват в лицето на играча и му отнемат една или повече точки живот;
	+ Лечебни отвари, които играчът изпива, при което му се възстановяват една или повече точки живот;
* Играчът да започва играта с максимален брой точки живот, като текущия му брой точки не може да надвишава този максимум;
* Изходът на всяко ниво е само един и при стъпването върху него играчът трябва да получава възможност да избере дали иска да премине в следващото ниво или предпочита да остане в текущото ниво и да се върне по-късно;
* Играчът може да се движи в четирите посоки с фиксирана скорост (например не повече от 1 квадратче на всеки 500 милисекунди);
* В началото на всяко ниво, играчът започва с фиксирано време, например 60 секунди; оставащото време при преминаване на следващо ниво не се прибавя към времето за преминаване на следващото ниво;
* При преминаване на следващо ниво, играчът запазва събрания до момента резултат от точки, както и текущия си наличен брой точки живот;
* Във всяко следващо ниво, съотношението на предмети с положителен ефект върху играча спрямо тези с отрицателен намалява (бомбите стават повече, отварите – по-малко);
* През цялото време, докато играе, играчът трябва да има информация за оставащото му време, както и за оставащия брой точки, бомби, отвари и други сходни предмети, които са скрити във все още непреровените контейнери в текущото ниво, така че да може да прецени кога да премине на следващото;
* Използвайте следните символи за плочки в игралното поле:
	+ ☺ (U+263A) – за играча;
	+ ▓ (U+2593) – за стена;
	+ ☼ (U+263C – за колона;
	+ ▲ (U+25B2) – за купчина с боклуци;
	+ ■ (U+25A0) – за сандък;
	+ ► (U+25BA) – за изхода към следващото ниво.

## Технически изисквания

* Да се използва обектно-ориентирано програмиране;
* Да се реализира клас за ниво от играта;
* Да се реализира клас за играча;
* Да се реализират базов клас за плочка от игралното поле и класове за различните видове препятствия, различните видове контейнери и изход към следващото ниво;
* Да се реализират базов клас за предмет, който може да се съдържа в контейнер, и класове за бомба, лечебна отвара и различните видове скъпоценности;
* Да се реализира интерфейс за контейнери, които могат да бъдат преравяни, с метод за преравянето на съответния контейнер;
* Да се реализира интерфейс за предмет, който се активира при намирането му (при преравяне на контейнер), с метод за активирането на съответния предмет, например избухването на бомба;
* Да се капсулират коректно всички обекти; използвайте свойства, където е уместно;
* Да се използват наследяване и полиморфизъм (виртуални членове, абстракция, интерфейси);
* За да се минимизират нежеланите премигвания на екрана, на всеки кадър екранът **не трябва** да се изчиства, а само да се препечатват необходимите символи на съответните места;
* Да се използват подходящи типове данни, конструкции за условни преходи и цикли;
* Да се разбие кодът на програмата на уместни методи;
* Да се документира програмата с подходящи коментари;
* Да се форматира кодът на програмата по прегледен начин и да се използват уместни идентификатори.

## Напътствия

* Използвайте двумерен масив с елементи от съответния базов клас за реализацията на игралното поле в нивото;
* Базовият клас за плочка от игралното поле да има член за символа, с който се представя плочката в конзолата;
* Реализирайте помощен клас за действие, което играчът извършва в момента (характеризиращо се с описание, продължителност и момент, в който е започнало) и направете метода на интерфейса за контейнери, които могат да бъдат преравяни, така, че да връща като резултат такова действие;
* Когато играчът прерови контейнер или се очаква от него да направи избор, спирайте брояча на време за нивото и извеждайте съответната информация за конзолата; при потвърждаване от страна на играча, отново покажете игралното поле и активирайте брояча.

# Допълнителни условия (за ентусиасти)

* Да се добавят допълнителни видове предмети, които могат да се съдържат в контейнерите, например:
	+ Капани, които забавят играча или напълно го обездвижват за определено време);
	+ Отвари, които увеличават скоростта на играча за определено време;
	+ Безполезни предмети;
* Играчът да има раница, в която да събира предмети (само събираеми такива). Тези предмети той да може да използва по-късно или да изхвърля (в случай, че раницата му се напълни). Колкото по-пълна е раницата, толкова по-бавен става играчът;
* Да се добавят движещи се препятствия (може да се движат на случаен принцип или пък да осцилират по определен начин);
* При генерирането на препятствия в нивото, да се използва алгоритъм за генериране на лабиринти;
* Скоростта на играча да е зависима от текущо наличния брой точки живот;
* Да се намалят до минимум прекъсванията на играта за извеждане на съобщения и изчакване на избор от страна на потребителя, като информацията се извежда в подходящ съкратен вид под игралното поле;
* Да се използват подходящи събития (например раняване и смърт на играча).