Қазақстан Республикасы Ауыл шаруашылығы министрлігі

С.Сейфуллин атындағы қазақ агротехникалық университеті

"Ақпараттық жүйелер" кафедрасы

КУРСТЫҚ ЖҰМЫС

Тақырыбы: "Python тілінде мәтінді түзететін бағдарлама құру"

Пәні: Алгоритм, деректер құрылымы және бағдарламалау

Орындаған: 23 – 06 тобының студенті

Аман Ерсұлтан Ерболұлы

Тексерген: Найзағараева А.А.

Курстық жұмыс қорғауға жіберілді

\_\_\_\_\_\_\_\_\_\_\_\_

(оқытушы қолы)

**Астана – 2024**

«С. СЕЙФУЛЛИН АТЫНДАҒЫ ҚАЗАҚ АГРОТЕХНИКАЛЫҚ ЗЕРТТЕУ УНИВЕРСИТЕТІ» КеАҚ

Факультеті КЖжКББ​

Кафедрасы Ақпараттық жүйелер​

БББ В05-057-21-01 ​

Курстық жұмысты орындауға

ТАПСЫРМА

Студент Зарлық Рауан Даниярұлы​

Жұмыстың тақырыбы Python тіліндегі мәтінде түзететін бағдарлама құру​

Жұмысқа бастапқы мәліметтер C++ тіліндегі мәтінде түзететін бағдарлама жазу. Бағдарлама тыныс белгісінің алдындағы бос орынды алып тастап, тыныс белгісінен кейін және сөздер арасында бір бос орын қойып, сөйлемнің бірінші әрпін бас әріпке түзетуі керек.

​

|  |  |  |  |
| --- | --- | --- | --- |
| № | Түсіндірме жазбаның мазмұны | Орындалу мерзімі | Шамамен көлем |
| 1 | Кіріспе | 12.10-13.10 | 1 бет |
| 2 | Алгоритм мен бағдарламалау | 13.10-18.10 |  |
| 3 | C++ бағдарламалау тілі | 18.10-24.10 |  |
| 4 | Data Structures and Algorithm Analysis in C++ | 24.10-28.10 |  |
| 5 | Тапсырма бойынша есептерді қарастыру | 28.10-31.10 |  |
| 6 | Қортынды | 31.10-01.11 |  |

Ұсынылатын негізгі әдебиеттер :

1. Б. Страуструп. Программалау: С++ тілімен танысу.2020 ж.

2. Н. Ж. Сарбаев. Мәліметтер құрылымы және алгоритмдер С++ тілінде. 2021ж.

3. М. Доран. Beginning C++ Game Programming. Packt Publishing, 2016 ж.

Тапсырманың берілген күні 23.09.2024

Жұмысты қорғау уақыты

Ғылыми жетекші ​​​​Найзағараева А.А.

Тапсырманы орындауға қабылдаған студент Аман Е.Е.

**Мазмұны**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Кіріспе . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | |  |
|  | Бағдарлама құрылымы мен жұмыс принциптері. . . . . . . . . . . . . . . . . | |  |
|  | Функциялардың сипаттамасы. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | |  |
|  | Бағдарламаны әзірлеу және іске қосу. . . . . . . . . . . . . . . . . . . . . . . . . . . . | |  |
|  | Бағдарламаны жетілдіру жолдары . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | |  |
|  | Алгоритмдерді оңтайландыру . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | |  |
|  | Ойын нәтижелері мен статистикасы . . . . . . . . . . . . . . . . . . . . . . . . . . . . | |  |
| Ойын нәтижелері мен статистикасы . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | |  |
| Бағдарламаның құрылымы мен сапасын жақсарту. . . . . . . . . . . . . . . . . | |  | |
| Қорытынды . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | |  |
| Пайдаланылған әдебиеттер тізімі. . . . . . . | | |  |

### Кіріспе

Бұл бағдарлама — лазердің қозғалысын, айналардың әсерін және нысаналарды жоюды модельдейтін қарапайым интерактивті ойын. Ойынның негізгі идеясы — лазердің белгілі бір бағытта қозғалуын бақылап, оның нысаналарды соғуын қамтамасыз ету. Алаңда орналасқан айналар лазердің бағытын өзгертеді, бұл ойынның стратегиясы мен күрделілігін арттырады.

**Негізгі ойын элементтері**:

* **Алаң**: Торлы алаң (10x10) нысаналар мен айналарды кездейсоқ түрде орналастырады.
* **Лазер**: Белгілі бір бастапқы нүктеден қозғала бастайды және айналар арқылы бағытын өзгертеді.
* **Нысаналар**: Лазер тиген кезде жойылады.
* **Айналар**: Лазердің бағытын өзгертетін элементтер.

Бағдарламаның мақсаты — барлық нысаналарды жойып, лазер қозғалысын визуалды түрде көрсету.

**Қолдану ерекшеліктері**:

1. Ойын алаңы толығымен кездейсоқ түрде қалыптасады, бұл әрбір іске қосуды ерекше етеді.
2. Бағдарлама лазердің қозғалысын, нысаналардың жойылуын және ойын алаңын әр қадам сайын көрсетеді.
3. Ойын лазердің алаңнан шыққан сәтінде немесе барлық нысаналар жойылғанда аяқталады.

Бұл бағдарлама ойыншыларға қызықты әрі қарапайым түрде логикалық ойлау қабілетін сынауға мүмкіндік береді.

Тақырып таңдау және бекіту

### Тақырып: ****Лазер мен айналардың өзара әрекеттесуін модельдеу: интерактивті ойын симуляциясының әзірленуі және жүзеге асырылуы****

Бұл тақырып бағдарламаның негізгі идеясын толық сипаттайды. Бағдарлама лазердің қозғалысын және айналармен өзара әрекеттесуін модельдейді, бұл ойын процесін қызықты әрі логикалық тұрғыда күрделендіреді. Сонымен қатар, нысаналарды жою арқылы ойынның мақсаты айқындалады, ал кездейсоқ түрде жасалған алаң әрбір іске қосуды ерекше әрі қайталанбас етеді.

**Жетекшілік жасау**

Бұл бағдарламаны түсіну және дамыту үшін оның логикасы, құрылымы мен функционалдық элементтерін жан-жақты қарастыру қажет. Бағдарлама лазердің қозғалысын, айналардың әсерін және нысаналардың жойылуын симуляциялайды. Әрі қарай, ойынның негізгі компоненттері мен алгоритмдерін егжей-тегжейлі түсіндіріп, бағдарламаны жетілдіру жолдарын талқылаймыз.

### 1. ****Бағдарламаның жалпы құрылымы****

Бағдарлама C++ тілінде құрылымдалған және бірнеше негізгі компоненттен тұрады:

* **Алаңның торы**: Ойынның негізгі аймағы, екі өлшемді массив ретінде беріледі.
* **Лазер**: Алаңда қозғалып, нысаналарды жоюды мақсат етеді.
* **Айналар**: Лазер бағытын өзгертетін элементтер.
* **Нысаналар**: Лазер тиген кезде жойылатын элементтер.

Бұл құрылым ойынның динамикалық сипатын қалыптастырады.

### 2. ****Кодтың жұмыс логикасы****

#### Алаңның дайындалуы (initializeField)

Бұл функция ойын алаңын құру үшін қолданылады. Оның негізгі сатылары:

1. **Бос алаң құру**:  
   Барлық ұяшықтардың бастапқы күйі бос деп анықталады (mirror = NONE, target = false).
2. **Нысаналарды орналастыру**:  
   Нысаналар кездейсоқ координаттарға қойылады. Егер таңдалған ұяшық бос емес болса, жаңа координат таңдалады.
3. **Айналарды орналастыру**:  
   Айналар да кездейсоқ координаттарға қойылады, бірақ олар нысаналар орналасқан ұяшықтарға қойылмайды.

#### Лазер қозғалысы (moveLaser)

Бұл функция лазердің қозғалысын және алаңдағы элементтермен өзара әрекетін басқарады:

1. **Қозғалыс бағыты**:  
   Лазердің ағымдағы координатасы (x, y) бағыты (dx, dy) бойынша өзгереді.
2. **Шекарадан шығу**:  
   Лазер алаң шегінен шыққанда, қозғалыс тоқтатылады.
3. **Нысаналарды жою**:  
   Егер лазер нысанаға тисе, сол нысана жойылады (target = false).
4. **Айналармен өзара әрекет**:  
   Лазер айнаға тисе, оның бағыты өзгереді:
   * LEFT\_SLASH (\): Лазердің бағытын келесі түрде өзгертеді:

dx және dy орындарын ауыстырады, dy теріс мәнге өзгереді.

* + RIGHT\_SLASH (/): Лазердің бағытын келесі түрде өзгертеді:

dx және dy орындарын ауыстырады, dx теріс мәнге өзгереді.

#### Алаңды көрсету (printField)

Бұл функция ойын алаңын графикалық түрде бейнелейді:

* **L**: Лазердің ағымдағы орны.
* **T**: Нысана.
* **\ және /**: Айналар.
* **.**: Бос ұяшық.

Ойыншы лазердің қозғалысын және алаңдағы өзгерістерді визуалды түрде бақылай алады.

### 3. ****Бағдарламаны жетілдіру жолдары****

#### 3.1. Қосымша функционалдық мүмкіндіктер

Бағдарламаны интерактивті әрі қызықты ету үшін келесі мүмкіндіктерді қосуға болады:

* **Қолмен айналар мен нысаналарды орналастыру**:  
  Ойыншыларға алаңды өздері жобалау мүмкіндігін беру.
* **Күрделілік деңгейі**:  
  Алаң өлшемін және айналар/нысаналар санын таңдау мүмкіндігі.
* **Уақыт шектеуі**:  
  Әрбір қозғалыс үшін шектеулі уақыт қосып, ойынға динамика енгізу.

#### 3.2. Бағытты көрсету

Лазердің қозғалыс бағытын графикалық түрде белгілеу. Мысалы,

* Оңға → >
* Төменге ↓ v
* Жоғары ↑ ^
* Солға ← <

#### 3.3. Графикалық интерфейс

Консольдік визуализацияны графикалық интерфейске ауыстыру үшін SFML немесе SDL сияқты кітапханаларды қолдану.

#### 3.4. Лазердің энергиясы

Лазердің белгілі бір қадам санына ғана жететін энергиясын шектеу. Бұл ойынға стратегиялық шешімдер енгізеді.

### 4. ****Алгоритмдердің тиімділігін арттыру****

#### Кодты оңтайландыру

1. Кездейсоқ координаттарды генерациялауды жылдамдату:
   * Қайта тексеру санын азайту үшін бос ұяшықтарды алдын ала санап алу.
2. Айналармен өзара әрекеттесуді нақтырақ алгоритммен орындау:
   * Лазер бағытын өзгерту есептеулерін матрицалық түрге келтіру.

#### Үлкен өлшемдегі алаң

Алаң өлшемін арттырғанда өнімділікті қамтамасыз ету үшін қажетсіз ұяшықтарға лазердің "соққы" есептеулерін жүргізбеу.

### 5. ****Ойынның нәтижесі мен статистикасы****

Ойын соңында ойыншыға нәтижелерді көрсету:

* Жойылған нысаналар саны.
* Лазердің қозғалыс саны.
* Қалған нысаналар.

Бұл ойыншыларға өз нәтижелерін бақылауға және келесі ойындарға стратегиялық жоспар құруға көмектеседі.

### 6. ****Кодтың құрылымы мен сапасын жақсарту****

#### Құрылымдық жақсартулар

* Лазерді, айналарды және нысаналарды жеке сыныптар (class) ретінде ұйымдастыру.
* initializeField, printField, және moveLaser функцияларын бөлшектеу.

#### Тест жазу

Бағдарламаның әрбір бөлігі үшін модульдік тесттер құрып, функционалдылықтың тұрақтылығын тексеру.

### Бағдарламаны дайындау

Бұл бөлімде бағдарламаны толық іске қосу, құру және оны жетілдіру үшін қажетті барлық қадамдарды түсіндіреміз. Әрбір қадамды егжей-тегжейлі талқылаймыз.

### 1. ****Қажетті құралдар мен орта****

Бағдарламаны жазу және іске қосу үшін қажетті құралдар:

#### 1.1. ****Компилятор мен құралдар****

C++ тілінде бағдарламаларды компиляциялау үшін компилятор қажет. Компиляторды таңдау сіздің операционды жүйеңізге байланысты болады:

* **Windows**:
  + **MinGW** немесе **MSVC** (Microsoft Visual Studio) қолдануға болады.
  + **MinGW** — бұл GCC компиляторының Windows жүйесі үшін арналған нұсқасы.
  + **MSVC** — Microsoft Visual Studio бағдарламалау ортасының бөлігі болып табылады, сондықтан ол Windows жүйесіне бейімделген.
* **Linux**:
  + **GCC** (GNU Compiler Collection) — Linux-та кеңінен қолданылатын компилятор.
  + **Clang** — GCC-ге балама компилятор, әсіресе macOS жүйесінде танымал.
* **MacOS**:
  + **Xcode Command Line Tools** орнатқаннан кейін, **Clang** компиляторы қолданылады.

#### 1.2. ****IDE немесе мәтіндік редактор****

Бағдарлама жазу үшін IDE (Integrated Development Environment) немесе қарапайым мәтіндік редактор қажет:

* **Visual Studio Code**: жеңіл әрі кеңейтілген редактор, көптеген плагиндермен жұмыс істейді.
* **Code::Blocks**: C++ бағдарламалау үшін арнайы құрылымдалған IDE.
* **CLion**: JetBrains компаниясынан кәсіби IDE.
* **Visual Studio**: Егер Windows жүйесінде жұмыс істеп жатсаңыз, бұл IDE өте жақсы жұмыс істейді.

### 2. ****Бағдарлама құрылымын түсіну****

Бағдарлама лазердің қозғалысын, айналардың әсерін және нысаналардың жойылуын модельдейді. Бұл бағдарламаның негізгі функционалды бөліктерін түсіну маңызды. Атап айтқанда:

#### 2.1. ****Өзгермелі айнымалылар мен деректер құрылымдары****

* **FIELD\_SIZE**: Ойын алаңының өлшемі. Бұл мән 10x10 деп белгіленген.
* **MirrorType**: Айналардың түрін анықтайтын мәндер (LEFT\_SLASH, RIGHT\_SLASH, NONE).
* **Cell**: Ұяшықтың сипаттамасы, онда айна мен нысана туралы ақпарат сақталады.
* **Laser**: Лазердің қазіргі жағдайы (орны және бағыты).
* **numTargets** және **numMirrors**: Нысаналар мен айналардың саны.

#### 2.2. ****Функциялар****:

* **initializeField**: Бұл функция алаңды бастапқы күйінде орнатады, нысаналар мен айналарды орналастырады.
* **printField**: Алаңның ағымдағы жағдайын шығарады, лазердің орнын, айналар мен нысаналарды көрсетеді.
* **moveLaser**: Лазердің қозғалысын басқарушы функция, айналар мен нысаналарды есепке алады.

### 3. ****Бағдарламаны әзірлеу****

#### 3.1. ****Алаңды құру****

Алаңды initializeField функциясы арқылы құру керек. Бұл функцияның ішінде барлық ұяшықтар бастапқыда бос болып орнатылады (яғни, айна жоқ және нысана жоқ). Кейін біз **нысаналарды** және **айналарды** кездейсоқ орналастырамыз.

cpp

Копировать код

void initializeField(vector<vector<Cell>>& field, int numTargets, int numMirrors) {

// Барлық ұяшықтарды бос қылу

for (int i = 0; i < FIELD\_SIZE; ++i) {

for (int j = 0; j < FIELD\_SIZE; ++j) {

field[i][j].mirror = NONE;

field[i][j].target = false;

}

}

// Нысаналарды кездейсоқ орналастыру

srand(time(0)); // Рандомизация үшін уақытты негіз етіп аламыз

for (int i = 0; i < numTargets; ++i) {

int x, y;

do {

x = rand() % FIELD\_SIZE;

y = rand() % FIELD\_SIZE;

} while (field[x][y].target); // Егер ұяшықта нысана бар болса, жаңа ұяшық іздеу

field[x][y].target = true;

}

// Айналарды кездейсоқ орналастыру

for (int i = 0; i < numMirrors; ++i) {

int x, y;

do {

x = rand() % FIELD\_SIZE;

y = rand() % FIELD\_SIZE;

} while (field[x][y].mirror != NONE || field[x][y].target);

field[x][y].mirror = (rand() % 2 == 0) ? LEFT\_SLASH : RIGHT\_SLASH;

}

}

#### 3.2. ****Лазердің қозғалысын басқару****

Лазердің қозғалысы бағдарламаның орталық бөлігі болып табылады. Лазер алаңда өзінің бағыты бойынша қозғалып, айналарды соғып, нысаналарды жояды. moveLaser функциясы бұл процесті басқарады.

bool moveLaser(vector<vector<Cell>>& field, Laser& laser) {

while (true) {

laser.x += laser.dx;

laser.y += laser.dy;

// Алаң шегінен шығу

if (laser.x < 0 || laser.x >= FIELD\_SIZE || laser.y < 0 || laser.y >= FIELD\_SIZE) {

return false;

}

// Нысананы соғу

if (field[laser.x][laser.y].target) {

field[laser.x][laser.y].target = false; // Нысананы жою

return true;

}

// Айнаны соғу

if (field[laser.x][laser.y].mirror == LEFT\_SLASH) {

// Бағытты өзгерту

swap(laser.dx, laser.dy);

laser.dy = -laser.dy;

} else if (field[laser.x][laser.y].mirror == RIGHT\_SLASH) {

// Бағытты өзгерту

swap(laser.dx, laser.dy);

laser.dx = -laser.dx;

}

}

}

#### 3.3. ****Алаңды бейнелеу****

Лазердің, айналардың және нысаналардың орналасуын көрсету үшін printField функциясы қолданылады.

void printField(const vector<vector<Cell>>& field, Laser laser) {

for (int i = 0; i < FIELD\_SIZE; ++i) {

for (int j = 0; j < FIELD\_SIZE; ++j) {

if (i == laser.x && j == laser.y)

cout << "L "; // Лазер

else if (field[i][j].target)

cout << "T "; // Нысана

else if (field[i][j].mirror == LEFT\_SLASH)

cout << "\\ "; // Сол жақ айна

else if (field[i][j].mirror == RIGHT\_SLASH)

cout << "/ "; // Оң жақ айна

else

cout << ". "; // Бос ұяшық

}

cout << endl;

}

cout << endl;

}

### 4. ****Бағдарламаны іске қосу****

#### 4.1. ****main функциясы****

Бағдарлама іске қосылмас бұрын, негізгі функцияда барлық компоненттерді біріктіру қажет.

int main() {

vector<vector<Cell>> field(FIELD\_SIZE, vector<Cell>(FIELD\_SIZE));

initializeField(field, 3, 5); // 3 нысана, 5 айна

Laser laser = {0, 0, 0, 1}; // Лазер бастапқыда (0,0) координатасынан оңға қарай қозғалады

cout << "Бастапқы алаң:" << endl;

printField(field, laser);

while (true) {

if (moveLaser(field, laser)) {

cout << "Нысана жойылды!" << endl;

} else {

cout << "Лазер алаңнан шықты!" << endl;

break;

}

printField(field, laser);

}

cout << "Ойын аяқталды!" << endl;

return 0;

}

Бұл функцияда алаңды жасаймыз, лазерді орналастырамыз және оның қозғалысын бақылаймыз.

### Бағдарламаның құрылымы

Бағдарламаның құрылымы — оның әрбір бөлігінің ұйымдастырылуын және жұмыс істеуін түсінуге көмектесетін маңызды аспект. Осы бағдарламаны құру кезінде оның құрылымын бірнеше бөлікке бөлуге болады.

#### 1. ****Бағдарлама құрылымының жалпы көрінісі****

Бұл бағдарлама **C++** тілінде жазылған, және оның құрылымы негізінен келесі бөліктерден тұрады:

1. **Жалпы анықтамалар**: Барлық қажетті кітапханалар мен анықтамалар.
2. **Деректер құрылымдары**: Ойын алаңы мен лазердің қозғалысын басқару үшін қажетті деректер.
3. **Функциялар**: Ойынның жұмысын ұйымдастыратын негізгі функциялар.
4. **main функциясы**: Бағдарламаның негізгі атқарушы бөлігі.

#### 2. ****Жалпы анықтамалар****

Бұл бөлікте бағдарламада қолданылатын барлық кітапханалар мен константалар анықталады. Мысалы:

* **<iostream>**: Деректерді экранға шығару үшін қолданылады.
* **<vector>**: Векторды пайдалану үшін қажет.
* **<cstdlib>**: Кездейсоқ сандарды генерациялау үшін.
* **<ctime>**: Уақытпен жұмыс істеп, кездейсоқ сандардың әрқашан өзгеріп тұруын қамтамасыз ету үшін.
* **<cmath>**: Егер қажет болса, математикалық операцияларды орындауға мүмкіндік береді.

**Константалар**:

* **FIELD\_SIZE**: Ойын алаңының өлшемін белгілейді, яғни 10x10 ұяшық.
* **MirrorType**: Айналардың түрлерін сипаттайтын **enum**.

#include <iostream>

#include <vector>

#include <cstdlib>

#include <ctime>

#include <cmath>

using namespace std;

const int FIELD\_SIZE = 10; // Ойын алаңының өлшемі

#### 3. ****Деректер құрылымдары****

**Деректер құрылымдары** бағдарламаның жүріп-тұруын ұйымдастырады. Мысалы:

* **MirrorType**: Айналардың типін сақтау үшін **enum** қолданылды.
  + **NONE**: Ұяшықта айна жоқ.
  + **LEFT\_SLASH**: Сол жақ қисық айна.
  + **RIGHT\_SLASH**: Оң жақ қисық айна.
* **Cell**: Ойын алаңындағы ұяшықтың сипаттамасын сақтайтын құрылым. Әр ұяшықта айна мен нысананың болуы мүмкін.

enum MirrorType { NONE, LEFT\_SLASH, RIGHT\_SLASH };

struct Cell {

MirrorType mirror;

bool target;

};

* **Laser**: Лазердің ағымдағы орнын және қозғалыс бағытын сипаттайтын құрылым.

cpp

Копировать код

struct Laser {

int x, y; // Лазердің координаталары

int dx, dy; // Лазердің қозғалыс векторы (бағыты)

};

#### 4. ****Функциялар****

Бағдарламаның әртүрлі бөлімдерін ұйымдастыру үшін функциялар пайдаланылған. Олардың негізгі мақсаттары — ойын алаңын құру, лазердің қозғалысын басқару және алаңның ағымдағы күйін көрсету.

* **initializeField**: Ойын алаңын бастапқы жағдайда құрады. Нысаналар мен айналарды кездейсоқ орналастырады.

void initializeField(vector<vector<Cell>>& field, int numTargets, int numMirrors) {

// Барлық ұяшықтарды бос қылу

for (int i = 0; i < FIELD\_SIZE; ++i) {

for (int j = 0; j < FIELD\_SIZE; ++j) {

field[i][j].mirror = NONE;

field[i][j].target = false;

}

}

// Нысаналарды және айналарды орналастыру

srand(time(0));

for (int i = 0; i < numTargets; ++i) {

int x, y;

do {

x = rand() % FIELD\_SIZE;

y = rand() % FIELD\_SIZE;

} while (field[x][y].target);

field[x][y].target = true;

}

for (int i = 0; i < numMirrors; ++i) {

int x, y;

do {

x = rand() % FIELD\_SIZE;

y = rand() % FIELD\_SIZE;

} while (field[x][y].mirror != NONE || field[x][y].target);

field[x][y].mirror = (rand() % 2 == 0) ? LEFT\_SLASH : RIGHT\_SLASH;

}

}

* **printField**: Бұл функция алаңды экранға шығарады, лазердің орнын, айналар мен нысаналарды көрсетеді.

void printField(const vector<vector<Cell>>& field, Laser laser) {

for (int i = 0; i < FIELD\_SIZE; ++i) {

for (int j = 0; j < FIELD\_SIZE; ++j) {

if (i == laser.x && j == laser.y)

cout << "L "; // Лазер

else if (field[i][j].target)

cout << "T "; // Нысана

else if (field[i][j].mirror == LEFT\_SLASH)

cout << "\\ "; // Сол жақ айна

else if (field[i][j].mirror == RIGHT\_SLASH)

cout << "/ "; // Оң жақ айна

else

cout << ". "; // Бос ұяшық

}

cout << endl;

}

cout << endl;

}

* **moveLaser**: Лазердің қозғалысын басқару. Лазердің бағыты айнаны соғып, нысананы жойып немесе алаңнан шығып кетуін қадағалайды.

bool moveLaser(vector<vector<Cell>>& field, Laser& laser) {

while (true) {

laser.x += laser.dx;

laser.y += laser.dy;

// Алаң шегінен шығу

if (laser.x < 0 || laser.x >= FIELD\_SIZE || laser.y < 0 || laser.y >= FIELD\_SIZE) {

return false;

}

// Нысананы соғу

if (field[laser.x][laser.y].target) {

field[laser.x][laser.y].target = false; // Нысананы жою

return true;

}

// Айнаны соғу

if (field[laser.x][laser.y].mirror == LEFT\_SLASH) {

// Бағытты өзгерту

swap(laser.dx, laser.dy);

laser.dy = -laser.dy;

} else if (field[laser.x][laser.y].mirror == RIGHT\_SLASH) {

// Бағытты өзгерту

swap(laser.dx, laser.dy);

laser.dx = -laser.dx;

}

}

}

#### 5. ****main функциясы****

**main()** функциясы бағдарламаның орындаушы бөлімі болып табылады. Мұнда алаңды инициализациялап, лазердің қозғалысын бақылаймыз, және ойынның нәтижелерін экранға шығарамыз.

int main() {

vector<vector<Cell>> field(FIELD\_SIZE, vector<Cell>(FIELD\_SIZE));

initializeField(field, 3, 5); // 3 нысана, 5 айна

Laser laser = {0, 0, 0, 1}; // Лазер бастапқыда (0,0) координатасынан оңға қарай қозғалады

cout << "Бастапқы алаң:" << endl;

printField(field, laser);

while (true) {

if (moveLaser(field, laser)) {

cout << "Нысана жойылды!" << endl;

} else {

cout << "Лазер алаңнан шықты!" << endl;

break;

}

printField(field, laser);

}

cout << "Ойын аяқталды!" << endl;

return 0;

}

### 6. ****Бағдарламаның жалпы құрылымы****

* **Мақсаты**: Лазер арқылы айналар мен нысаналарды өңдеу.
* **Деректер құрылымдары**: Ойын алаңының өлшемі мен объектілерінің типтері (айна, нысана, лазер).
* **Функциялар**: Әрбір компоненттің жұмысын басқару үшін қолданылатын жеке функциялар.
* **main функциясы**: Барлық процесті орындау және ойынның ағымдағы жағдайын шығару.

#### ****Бағдарламаның жалпы құрылымына арналған ресімдеме****

Бағдарламаның жалпы құрылымы дұрыс және түсінікті болуы үшін мынадай ережелерді сақтау керек:

1. **Кітапханаларды қосу**: Барлық кітапханалар бағдарламаның басында тұруы керек. Бұл бағдарламаның жақсы құрылымдалғанын және қандай кітапханалар қолданылатынын бірден көруге мүмкіндік береді. Кітапханалардың арасында қосымша бос орындар қалдырмау ұсынылады.

#include <iostream>

#include <vector>

#include <cstdlib>

#include <ctime>

#include <cmath>

1. **Константалар мен анықтамаларды түсініктемелермен ресімдеу**: Әрбір константа немесе айнымалының не үшін қолданылатынын түсіндіретін түсініктемелер қосу маңызды. Бұл әсіресе бағдарламаның масштабын ұлғайту кезінде пайдалы.

const int FIELD\_SIZE = 10; // Алаңның өлшемі 10x10

1. **Деректер құрылымдарына арналған дұрыс формат**: Әрбір құрылымның өз алдына орналасуын және түсініктемелерді көрсету маңызды. Бұл бағдарламаның оқылуын арттырады және оның ұқыптылығын көрсетеді.

enum MirrorType { NONE, LEFT\_SLASH, RIGHT\_SLASH };

struct Cell {

MirrorType mirror; // Айнаның типі

bool target; // Нысана бар ма?

};

struct Laser {

int x, y; // Лазердің координаталары

int dx, dy; // Лазердің бағыты (бағыт векторы)

};

#### 2. ****Функцияларды ресімдеу****

Әрбір функцияның міндеті анық көрсетілуі керек. Функцияның аты, параметрлері және оның не үшін қолданылатынын түсіндіретін комментарийлер болуы керек.

1. **initializeField функциясы**: Бұл функция алаңды бастапқы күйде инициализациялайды, нысаналарды және айналарды кездейсоқ орналастырады.

void initializeField(vector<vector<Cell>>& field, int numTargets, int numMirrors) {

for (int i = 0; i < FIELD\_SIZE; ++i) {

for (int j = 0; j < FIELD\_SIZE; ++j) {

field[i][j].mirror = NONE; // Айна жоқ

field[i][j].target = false; // Нысана жоқ

}

}

srand(time(0)); // Уақыт бойынша кездейсоқ сандар генерациясы

for (int i = 0; i < numTargets; ++i) {

int x, y;

do {

x = rand() % FIELD\_SIZE;

y = rand() % FIELD\_SIZE;

} while (field[x][y].target); // Егер нысана бар болса, басқа орын таңда

field[x][y].target = true; // Нысана орналастыру

}

for (int i = 0; i < numMirrors; ++i) {

int x, y;

do {

x = rand() % FIELD\_SIZE;

y = rand() % FIELD\_SIZE;

} while (field[x][y].mirror != NONE || field[x][y].target);

field[x][y].mirror = (rand() % 2 == 0) ? LEFT\_SLASH : RIGHT\_SLASH;

}

}

1. **printField функциясы**: Бұл функция алаңның ағымдағы күйін экранға шығарады, лазердің орнын, айналар мен нысаналарды көрсетеді.

void printField(const vector<vector<Cell>>& field, Laser laser) {

for (int i = 0; i < FIELD\_SIZE; ++i) {

for (int j = 0; j < FIELD\_SIZE; ++j) {

if (i == laser.x && j == laser.y)

cout << "L "; // Лазердің орны

else if (field[i][j].target)

cout << "T "; // Нысана

else if (field[i][j].mirror == LEFT\_SLASH)

cout << "\\ "; // Сол жақ қисық айна

else if (field[i][j].mirror == RIGHT\_SLASH)

cout << "/ "; // Оң жақ қисық айна

else

cout << ". "; // Бос ұяшық

}

cout << endl;

}

cout << endl;

}

1. **moveLaser функциясы**: Лазердің қозғалысы мен оның айналармен өзара әрекеттесуін басқару.

bool moveLaser(vector<vector<Cell>>& field, Laser& laser) {

while (true) {

laser.x += laser.dx;

laser.y += laser.dy;

// Алаң шегінен шығу

if (laser.x < 0 || laser.x >= FIELD\_SIZE || laser.y < 0 || laser.y >= FIELD\_SIZE) {

return false;

}

// Нысананы соғу

if (field[laser.x][laser.y].target) {

field[laser.x][laser.y].target = false;

return true;

}

// Айнаны соғу

if (field[laser.x][laser.y].mirror == LEFT\_SLASH) {

swap(laser.dx, laser.dy); // Бағытты өзгерту

laser.dy = -laser.dy;

} else if (field[laser.x][laser.y].mirror == RIGHT\_SLASH) {

swap(laser.dx, laser.dy); // Бағытты өзгерту

laser.dx = -laser.dx;

}

}

}

#### 3. main ****функциясының ресімдемесі****

**main()** функциясында барлық негізгі процестер жүзеге асырылады, сонымен қатар экранға нәтижелер көрсетіледі.

int main() {

vector<vector<Cell>> field(FIELD\_SIZE, vector<Cell>(FIELD\_SIZE));

initializeField(field, 3, 5); // 3 нысана, 5 айна

Laser laser = {0, 0, 0, 1}; // Лазер бастапқыда (0,0) координатасынан оңға қарай қозғалады

cout << "Бастапқы алаң:" << endl;

printField(field, laser);

while (true) {

if (moveLaser(field, laser)) {

cout << "Нысана жойылды!" << endl;

} else {

cout << "Лазер алаңнан шықты!" << endl;

break;

}

printField(field, laser);

}

cout << "Ойын аяқталды!" << endl;

return 0;

}

#### 4. ****Ресімдеу мен түсініктемелердің маңызы****

* **Кодты түсінікті ету**: Бағдарламаның әрбір бөлігін түсіндіретін комментарийлер қосу керек. Бұл бағдарламаны басқа бағдарламалаушыларға немесе пайдаланушыларға түсінікті етеді.

Пайдаланылған әдебиеттер

 **Bjarne Stroustrup, "The C++ Programming Language", 4th Edition**  
C++ тілін үйрену үшін негізгі оқулық. Бағдарламалауға қатысты барлық маңызды ұғымдар мен құрылымдарды қамтиды. Бұл кітап арқылы сіз бағдарламалау тілінің негіздерін түсінесіз.

 **Stanley B. Lippman, "C++ Primer", 5th Edition**  
C++ тілінің бастауыш деңгейден жоғары деңгейге дейінгі мүмкіндіктері мен ерекшеліктері туралы кітап. Жаңадан бастағандар үшін өте пайдалы.

 **Scott Meyers, "Effective C++: 55 Specific Ways to Improve Your Programs and Designs"**  
Бағдарламалауды тиімді әрі дұрыс жолмен жасау үшін бірнеше кеңестер. Бұл кітап бағдарламаңызды жетілдіруге және жақсырақ етуге көмектеседі.

 **Herb Sutter, "Exceptional C++"**  
C++ бағдарламалау тілінде өте жақсы және тиімді код жазуға арналған кеңестер мен мысалдар.

 **Robert Sedgewick, "Algorithms in C++, Part 1"**  
Бұл кітапта бағдарламалау үшін қажетті алгоритмдер мен деректер құрылымдары қарастырылады. Лазер ойындарындағы қозғалыс пен айналар сияқты логика үшін пайдалы.

 **Mark Allen Weiss, "Data Structures and Algorithm Analysis in C++"**  
Бұл кітапта деректер құрылымдары мен алгоритмдер туралы негізгі ақпарат берілген. Ойын алаңын құру кезінде қолданылатын құрылымдар мен логика үшін өте пайдалы.

 **C++ Reference (cppreference.com)**  
C++ стандартты кітапханасы туралы ақпарат беретін веб-сайт. Бағдарламалау кезінде жиі кездесетін функциялар мен командалар туралы толық мәлімет аласыз.

 **Jesse Liberty, "Programming C++"**  
C++ бағдарламалау тілін жақсы меңгеруге арналған кітап. Бағдарлама жасау кезінде пайдаланатын негізгі құралдар мен әдістерді түсінуге көмектеседі.

 **"The C++ Standard Library" by Nicolai M. Josuttis**  
C++ стандартты кітапханасын тереңірек түсіну үшін. Бұл кітапта жинақтар, файлдарды оқу-жазу және басқа да маңызды кітапхана құралдары қарастырылған.

 **"Beginning C++ Through Game Programming" by Michael Dawson**  
Ойын бағдарламалауға арналған C++ тілінің негіздері. Лазерлер, айна қозғалысы мен ойын логикасын жасаған кезде пайдалы болады.