ПРИДНЕСТРОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. Т.Г. ШЕВЧЕНКО

Рыбницкий филиал

Кафедра информатики и программной инженерии

НАУЧНО-ПРАКТИЧЕСКИЙ ПРОЕКТ

в рамках центра инженерных инициатив и образования, студенческого научного объединения «Информационно-коммуникационные практики и инициативы в регионе» кафедры ИиПИ

на тему:

«РАЗРАБОТКА ОБУЧАЮЩЕЙ ПРОГРАММЫ С ИГРОВЫМИ ЭЛЕМЕНТАМИ»

Студентов I курса направления
«Программная инженерия»
профиля «Разработка программно-информационных систем»
Корни Андрея Геннадьевича
Лаптурова Вадима Александровича

Научные руководители:
доцент, канд.экон. наук Тягульская Л.А.
старший преподаватель
Гарбузняк Елена Сергеевна

Рыбница, 2022

ВВЕДЕНИЕ

- **Актуальность** проекта определяется тем, что для решения проблемы, связанной с обучением в Интернете, поможет специальная программа, предназначенная для обучения пользователя. Поэтапное обучение поможет ученику не потеряться в потоке новой информации, а игровой подход к обучению будет ориентирован на молодую аудиторию, что придаст программе привлекательность.

- **Объект исследования** – обучающие программы

- **Предмет исследования** – игровые элементы для обучения программированию.

ЦЕЛЬ И ЗАДАЧИ

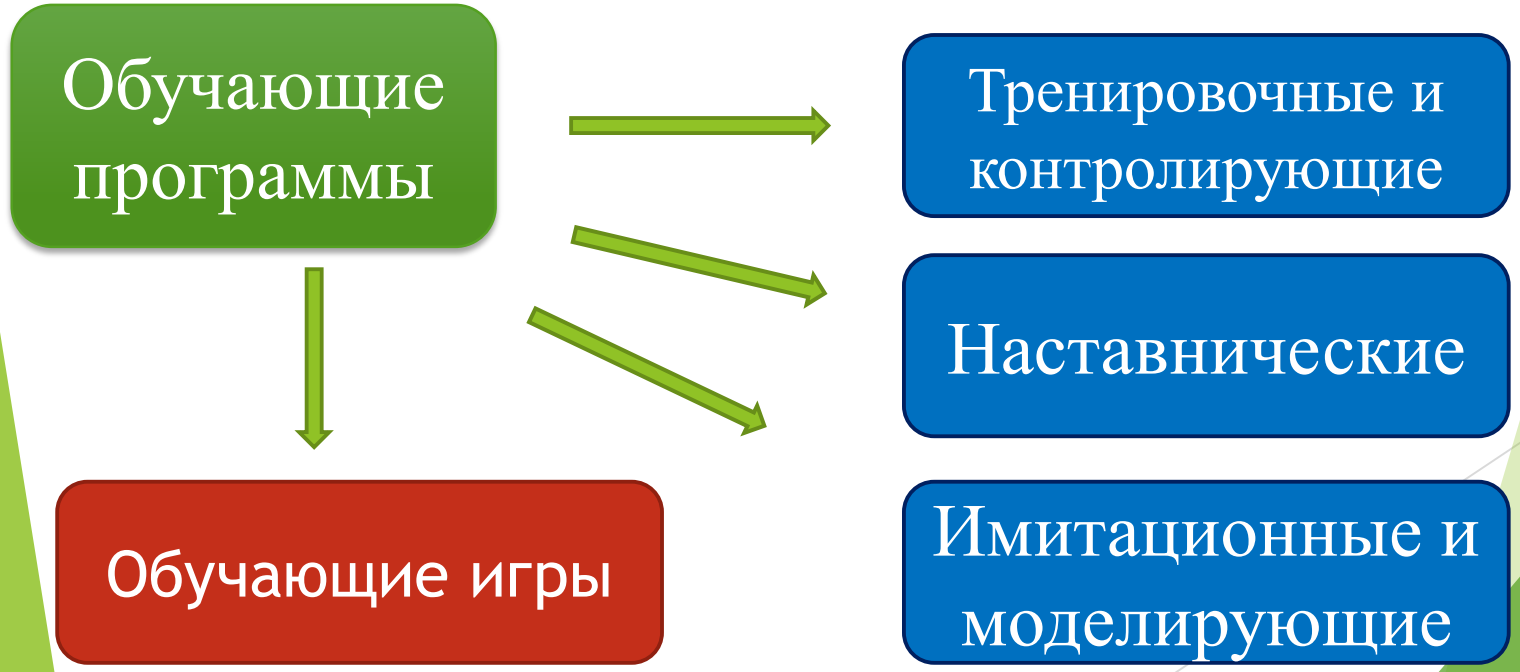
Целью проекта является создание обучающей программы с игровыми элементами.

Для достижения цели были поставлены следующие **задачи**:

- изучить понятие и виды обучающей программы и способы обучения программированию;
- рассмотреть понятие геймификации;
- рассмотреть игровые элементы в обучении;
- рассмотреть виды существующих обучающих программ и методов решений подобных задач;
- выбрать среду программирования;
- реализовать программный продукт.

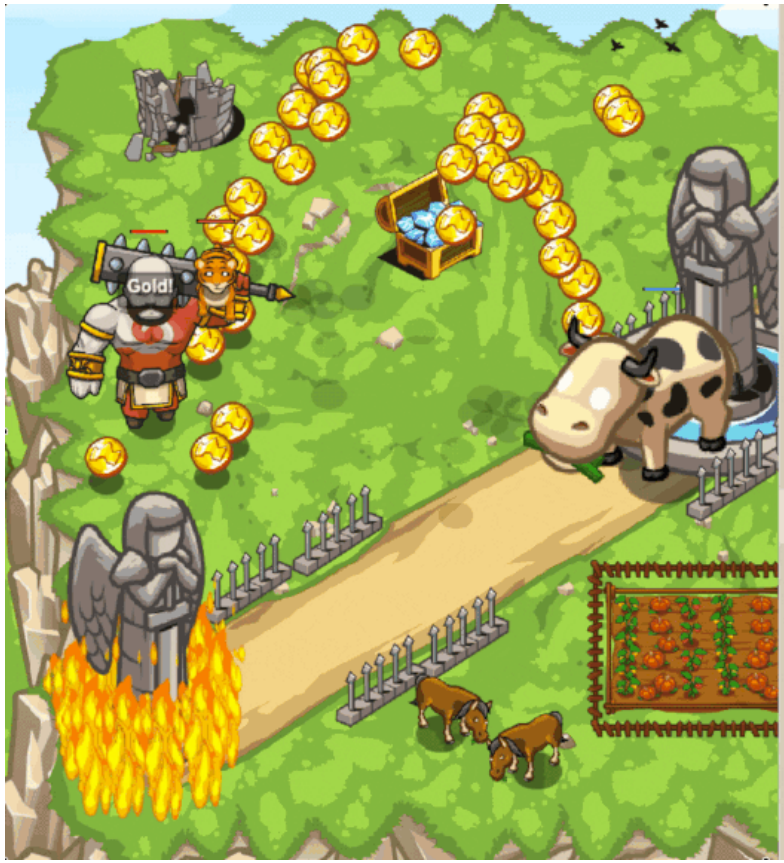
ПОНЯТИЕ И ВИДЫ ОБУЧАЮЩЕЙ ПРОГРАММЫ И СПОСОБЫ ОБУЧЕНИЯ ПРОГРАММИРОВАНИЮ

Обучающая программа - представляет собой учебное пособие, предназначенное для самостоятельной работы учащихся. Обучение должно способствовать максимальной активизации обучаемых, индивидуализируя их работу и предоставляя им возможность самим управлять своей познавательной деятельностью.



ГЕЙМИФИКАЦИЯ В ОБУЧЕНИИ

Основной способ привлечь и удержать обучаемую аудиторию – внедрить методику, сопутствующую обучению, которая и будет привлекать пользователей. Игровой способ обучения как раз способен на такое, специализируясь в основном на молодом возрасте.




```
1  hero = game.spawnPlayer("goliath")
2  boss = game.spawnXY("cow", 46, 40)
3  coin = game.spawnXY("gold", 0, 0)
4  boss.missileType = coin
5
6  game.score = 0
7
8  def onCollect(event):
9      if event.item.type == "gold":
10         event.target.say("Gold!")
11         game.score += item.value
12         if game.score > 100:
13             game.showVictory()
14
15 hero.on("collect", onCollect)
16
```

ИНСТРУМЕНТЫ РАЗРАБОТКИ

- Язык – C++

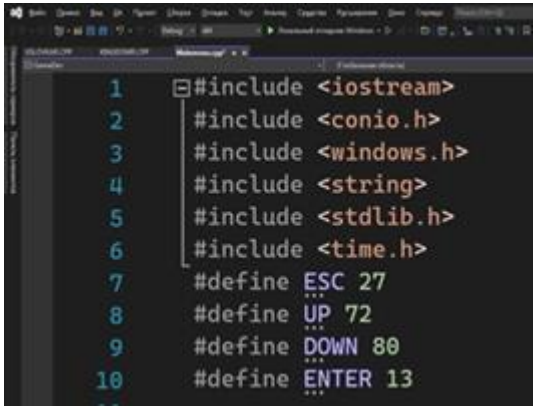
- Среда разработки - «*Visual Studio Code*» / Turbo C++

СОЗДАНИЕ ПРОГРАММЫ

Основные **этапы** создания программы:

- создание меню программы;
- создание игры «Snake»;
- создание обучающей программы-теста.

Подключение библиотек для создания меню



```
#include <iostream>
#include <conio.h>
#include <windows.h>
#include <string>
#include <stdlib.h>
#include <time.h>
#define ESC 27
#define UP 72
#define DOWN 80
#define ENTER 13
```

Цикл, выводящий на экран массив тем меню

```
int active_menu = 0;
char ch;
while (true)
{
    int x = 33, y = 10;
    GoToXY(x, y);

    for (int i = 0; i < size(Menu); i++)
    {
        if (i == active_menu)   SetConsoleTextAttribute(hStdOut, FOREGROUND_GREEN | FOREGROUND_INTENSITY);
        else    SetConsoleTextAttribute(hStdOut, FOREGROUND_GREEN);
        GoToXY(x, y++);
        cout << Menu[i] << endl;
    }
```

ВНУТРЕННИЙ ФУНКЦИОНАЛ ПРОГРАММЫ

Оператор множественного выбора для работы кнопок меню:

```
case ENTER:
    switch (active_menu)
    {
    case 0:
    case 1:
    case 2:
        system("CLS");
        GoToXY(x, y);
        SetConsoleTextAttribute(hStdOut, FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_INTENSITY);
        cout << "Пользователь выбрал \"" << Menu[active_menu] << "\"";
        _getch();
        system("CLS");
        break;
    case 3:
        exit(0);
    }
    break;
```

Процедура скрытия курсора

```
20  void ConsoleCursorVisible(bool show, short size)
21  {
22      CONSOLE_CURSOR_INFO structCursorInfo;
23      GetConsoleCursorInfo(hStdOut, &structCursorInfo);
24      structCursorInfo.bVisible = show; // изменяем видимость курсора
25      structCursorInfo.dwSize = size;   // изменяем размер курсора
26      SetConsoleCursorInfo(hStdOut, &structCursorInfo);
27  }
```

ВНУТРЕННИЙ ФУНКЦИОНАЛ ПРОГРАММЫ

Оператор множественного выбора switch, задача которого определять вводимый пользователем код клавиш клавиатуры

```
ch = _getch();
if (ch == -32) ch = _getch(); // Отлавливаем стрелочки
switch (ch)
{
case ESC:
    exit(0);
case UP:
    if (active_menu > 0)
        --active_menu;
    break;
case DOWN:
    if (active_menu < size(Menu) - 1)
        ++active_menu;
    break;
case ENTER:
    switch (active_menu)
    {
    case 0:
    case 1:
    case 2:
        system("CLS");
        GoToXY(x, y);
        SetConsoleTextAttribute(hStdOut, FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_INTENSITY);
        cout << "Пользователь выбрал \"" << Menu[active_menu] << "\"";
        _getch();
        system("CLS");
        break;
    case 3:
        exit(0);
    }
    break;


default:
    cout << "Код " << (int)ch << endl;
}
```

ВНУТРЕННИЙ ФУНКЦИОНАЛ ПРОГРАММЫ

Присваивание кнопкам значения для управления процессом передвижения

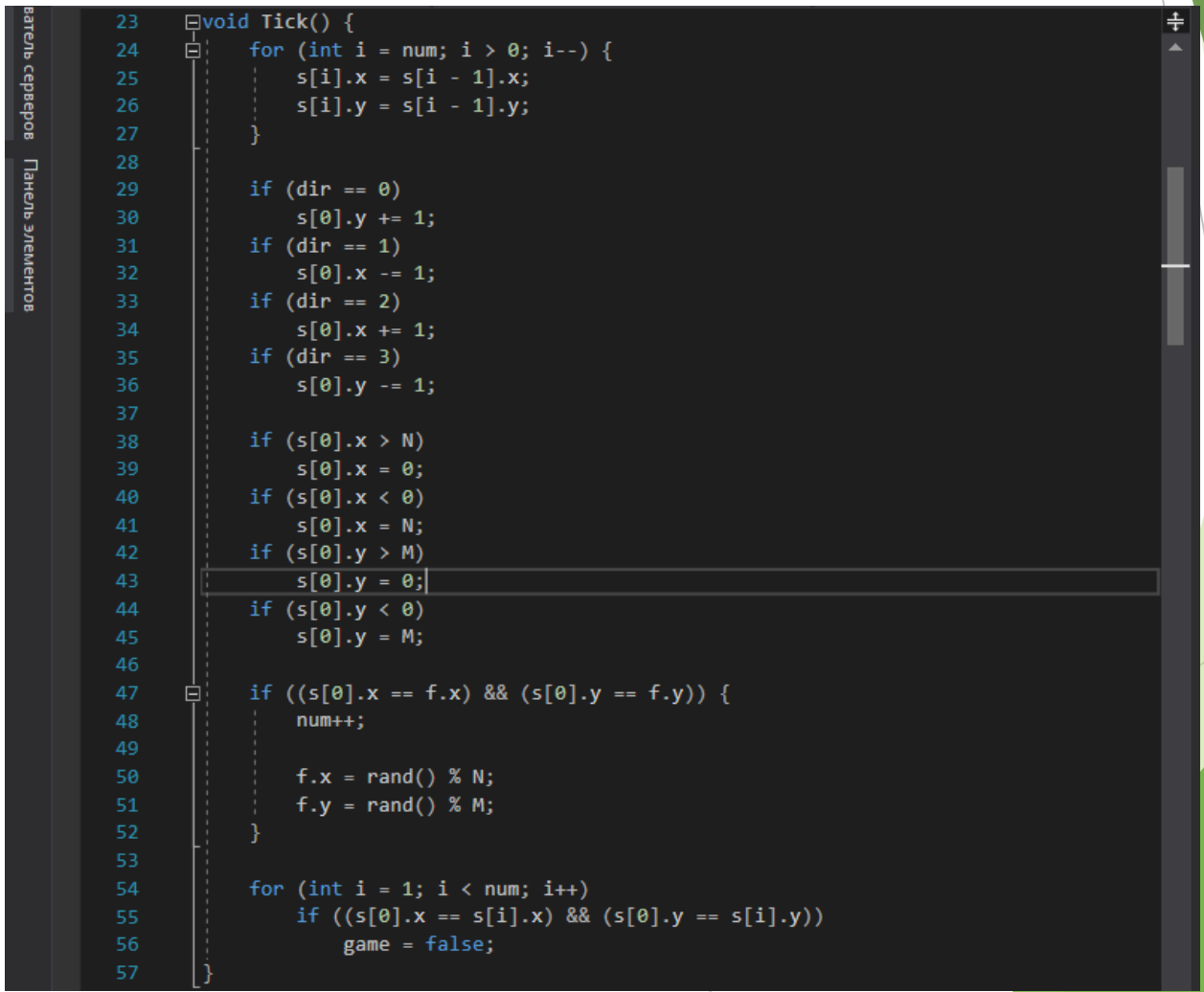
```
112         if (Keyboard::isKeyPressed(Keyboard::Left))
113             dir = 1;
114         if (Keyboard::isKeyPressed(Keyboard::Right))
115             dir = 2;
116         if (Keyboard::isKeyPressed(Keyboard::Up))
117             dir = 3;
118         if (Keyboard::isKeyPressed(Keyboard::Down))
119             dir = 0;
```

Структуры для координирования появления фруктов и самой змейки

```
14  struct Snake {
15      int x, y;
16  } s[600];
17
18  struct Fruct
19  {
20      int x, y;
21  } f;
```

ВНУТРЕННИЙ ФУНКЦИОНАЛ ПРОГРАММЫ

Процедура движения



```
23  void Tick() {
24      for (int i = num; i > 0; i--) {
25          s[i].x = s[i - 1].x;
26          s[i].y = s[i - 1].y;
27      }
28
29      if (dir == 0)
30          s[0].y += 1;
31      if (dir == 1)
32          s[0].x -= 1;
33      if (dir == 2)
34          s[0].x += 1;
35      if (dir == 3)
36          s[0].y -= 1;
37
38      if (s[0].x > N)
39          s[0].x = 0;
40      if (s[0].x < 0)
41          s[0].x = N;
42      if (s[0].y > M)
43          s[0].y = 0;
44      if (s[0].y < 0)
45          s[0].y = M;
46
47      if ((s[0].x == f.x) && (s[0].y == f.y)) {
48          num++;
49
50          f.x = rand() % N;
51          f.y = rand() % M;
52      }
53
54      for (int i = 1; i < num; i++)
55          if ((s[0].x == s[i].x) && (s[0].y == s[i].y))
56              game = false;
57  }
```

ВНУТРЕННИЙ ФУНКЦИОНАЛ ПРОГРАММЫ

Условие увеличения змейки и добавление позиции яблока

```
134     for (int i = 0; i < num; i++) {
135         if (i != 0)
136             snake.setTextureRect(IntRect(0, 0, ts, ts));
137         else
138             snake.setTextureRect(IntRect(dir*ts, ts, ts, ts));
139
140         if (!game && i == 1)
141             snake.setTextureRect(IntRect(dir * ts, ts * 2, ts, ts));
142
143         snake.setPosition(s[i].x * ts, s[i].y * ts);
144         window.draw(snake);
145     }
146
147     apple.setPosition(f.x * ts, f.y * ts);
148     window.draw(apple);
```

ВНУТРЕННИЙ ФУНКЦИОНАЛ ПРОГРАММЫ

Тест в текстовом формате в файле формата rtf

```
1 Класс - это:
0) любой тип данных, определяемый пользователем
1) * тип данных, определяемый пользователем и сочетающий в себе данные и
функции их обработки
2) структура, для которой в программе имеются функции работы с нею
2 Членами класса могут быть
0) * как переменные, так и функции, могут быть объявлены как private и
как public
1) только переменные, объявленные как private
2) только переменные и функции, объявленные как private
3 Что называется конструктором?
0) * метод, имя которого совпадает с именем класса и который вызывается
автоматически при создании объекта класса
1) метод, имя которого совпадает с именем класса и который вызывается
автоматически при объявлении класса (до создания объекта класса)
2) метод, имя которого необязательно совпадает с именем класса и который
вызывается при создании объекта класса
4 Объект - это
0) переменная, содержащая указатель на класс
1) * экземпляр класса
2) класс, который содержит в себе данные и методы их обработки
5 Что называется деструктором?
0) метод, который уничтожает объект
1) метод, который удаляет объект
2) * метод, который освобождает память, занимаемую объектом
```

ВНУТРЕННИЙ ФУНКЦИОНАЛ ПРОГРАММЫ

Считывание строк из файла и присвоение их массиву вопросов и ответов

```
for (int i = 0; i < SIZE; i++)
{
    getline(ifs, questions[i]);
    for (int j = 0; j < 3; j++)
    {
        getline(ifs, answers[i][j]);
    }

}
for (int i = 0; i < SIZE; i++)
{
    cout << questions[i].c_str() << endl;
    for (int j = 0; j < 3; j++)
    {
        cout << answers[i][j].c_str() <<endl;
    }
    cin >> ans;
```

ВНУТРЕННИЙ ФУНКЦИОНАЛ ПРОГРАММЫ

Проверка введенного варианта ответа

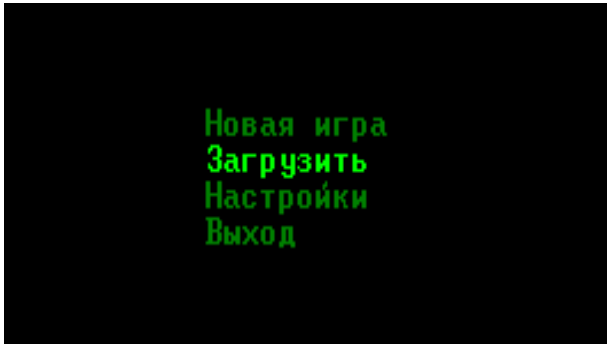
```
if (right_answer[i] == ans)
{
    ++cnt;
    cout << "This is the correct answer\n\n";
}
else
{
    cout << "This is the wrong answer\n\n";
}
```

Вывод количества правильных ответов и процента успешности

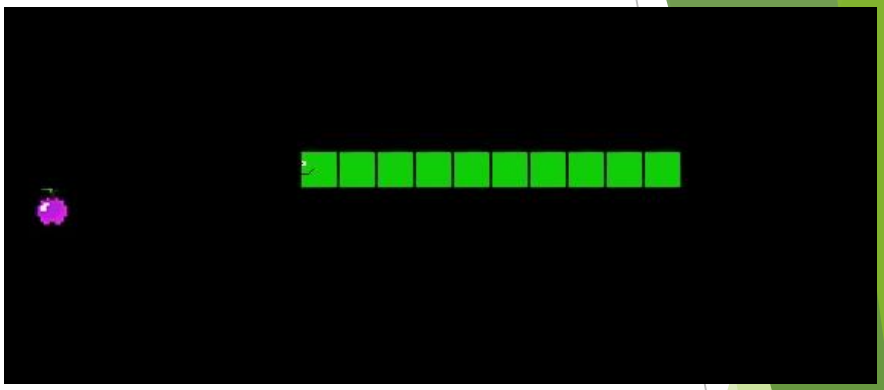
```
percent = ((float)cnt / SIZE) * 100;
cout << "Test results\nNumber of correct answers: " << cnt << endl;
cout << "Percentage of correct answers: " << percent << endl;
ifs.close();
system("pause");
```

ТЕСТИРОВАНИЕ

Запуск меню



При нажатии на кнопку «Новая игра» начинается новая игра



Прохождение теста при подборе трех яблок

```
------------------------------------------------------------
10 Полиморфизм - это :
0) средство, позволяющее в одном классе использовать методы с одинаковыми имена
и
1) * средство, позволяющее использовать одно имя для обозначения действий, общи
 для родственных классов
2) средство, позволяющее в одном классе использовать методы с разными именами д
я выполнения одинаковых действий
0
This is the wrong answer
```

ЗАКЛЮЧЕНИЕ

В ходе тестирования созданного программного продукта были выявлены следующие **недостатки**:

- ограниченность возможностей консольного меню;
- обучение операторам только одного языка.

При этом были отмечены такие **преимущества**, как:

- простота интерфейса;
- простота кода для ознакомления с тем, как работает обучающая программа.

ЗАКЛЮЧЕНИЕ

В перспективе в программу будут добавлены:

- игровые элементы;
- нескольких видов игровых решений для обучения;
- системы уровней и наград;
- счетчик прогресса;
- графические элементы;
- обучение другим языкам программирования;
- музыкальное сопровождение во время игры.

Подводя итоги, можно сказать, что разработанный программный продукт может быть полезен для детей младших классов, которые будут заинтересованы в программировании.