**Лабораторная работа 5. Приложение «Робот-Таракан»**

Динамическое создание компонентов

Приложение «Робот-Таракан» – приложение для изучения линейных алгоритмов (рис 1.). Робот-Таракан это исполнитель, в систему команд которого входят:

1. Команды изменения направления: Up, Down, Right, Left.
2. Команда перемещения на заданное расстояние – Step.

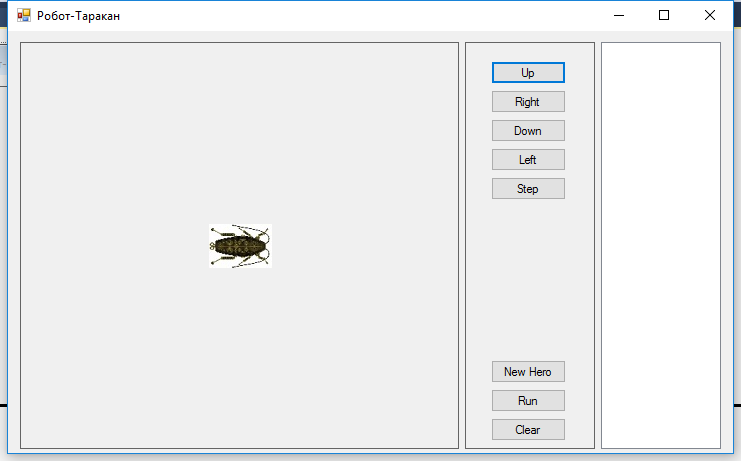


Рисунок 1. Поле программы

Меню работы с Тараканом состоит из кнопок с основными командами, при нажатии на которые пользователь может составить алгоритм для него.

Меню программы так же содержит кнопку для создания нового Таракана, кнопку запуска алгоритма и кнопку очистки поля алгоритма.

Одновременно в приложении могут находиться несколько Тараканов. Каждый Таракан может исполнять свой алгоритм. Исполнитель алгоритма выбирается нажатием мыши. Приложение позволяет перетаскивать Таракана по полю.

**Описание интерфейса**

Создайте новое решение с GUI-приложением. На форму поместите следующие компоненты:

|  |  |  |
| --- | --- | --- |
| **Компонент** | **Имя** | **Назначение компонента** |
| Panel | Field | поле для Таракана |
| Panel | Group | для группировки кнопок |
| ListBox | Algorithm | для записи алгоритма |
| Button | UpBtn,  RightBtn,  DownBtn,  LeftBtn,  StepBtn  NewBtn  RunBtn  ClearBtn | кнопки для выполнения Тараканом соответствующих команд изменения направления: вверх, направо, вниз, налево.  команда выполнения шага  создание нового Таракана  запуск алгоритма  очистка алгоритма |
| Timer | timerAlgorithm | таймер для дискретности алгоритма |

**Описание кода программы**

В пространстве имен созданного решения опишите переменную direction перечисление со значениями: Up, Right, Down, Left:

enum direction:byte { Up, Right, Down, Left };

Добавьте к проекту новый элемент – класс с названием Cockroach.

Для описания Таракана достаточно трех полей: изображения, направления и шага:

Bitmap Image;

direction trend = direction.Right;

const int step = 30;

Создайте конструктор для создания экземпляра класса с заданным изображение:

//Конструктор с параметром – изображение

public Cockroach(Bitmap \_Image)

Добавьте методы-свойства X и Y для хранения и вычисления координат Таракана на поле (координаты – целые чиса).

Опишите метод RePaint для отображения изображения в PictureBox. При повороте Таракана картинка будут изменяться размеры картинки, поэтому ее нужно перерисовать:

//Изображение объекта Таракан в PictureBox используется при изменении направления Таракана

public void RePaint(PictureBox p)

{

p.Bounds = new Rectangle(X, Y, Image.Width, Image.Height);//создание новых границ изображения для PictureBox

p.Image = Image;

}

Для отображения Таракана на поле можно описать метод Show:

//Изображение Таракана на поле

public void Show(PictureBox p, Panel owner)

{

X = (owner.Width - Image.Width) / 2;

Y = (owner.Height - Image.Height) / 2;

RePaint(p);

owner.Controls.Add(p);// добавляем PictureBox к элементу Panel

}

Метод выполнения шага в заданном направлении изменяет соответствующую координату на значение step в зависимости от направления:

public void Step()

{

switch (trend)

{

case direction.Right: ...

}

}

Метод изменения направления в зависимости от текущего направления:

//Изменение направления, параметр – первая буква направления

public void ChangeTrend(char c)

{

direction newtrend=trend;

for (direction y = direction.Up;y<=direction.Left; y++ )

if (char.ToLower(c) == char.ToLower(y.ToString()[0]))

{

newtrend = y;

break;

}

switch (trend)

{

case direction.Up:

switch (newtrend)

{

case direction.Right: Image.RotateFlip(RotateFlipType.Rotate90FlipNone); break;

case direction.Down: Image.RotateFlip(RotateFlipType.Rotate180FlipNone); break;

case direction.Left: Image.RotateFlip(RotateFlipType.Rotate270FlipNone); break;

}

break;

case direction.Right:

switch (newtrend)

{

}

break;

case direction.Down:

switch (newtrend)

{

}

break;

case direction.Left:

switch (newtrend)

{

}

break;

}

trend = newtrend;

}

}

Допишите самостоятельно метод ChangeTrend(char c)

**Приложение**

Для изображения Таракана на экране будем использовать компонент PictureBox. Обратите внимание, все объекты PictureBox будет в программе создаваться динамически, по требованию.

В классе формы создайте две переменные (поля):

Cockroach workCockroach;//рабочий Таракан - активный Таракан, который будет выполнять алгоритм

PictureBox workpb;//рабочее поле PictureBox - поле на котрой будет рабочий Таракан

List<Cockroach> LC;//Список для хранения созданных Тараканов

List<PictureBox> PB;//Список для хранения созданных объектов PictureBox

В конструкторе формы поместите

LC = new List<Cockroach>();

PB = new List<PictureBox>();

Обработчик нажатия кнопки NewBtn содержит следующий код:

Cockroach cockroach = new Cockroach(new Bitmap("cockroach1.jpg"));//

PictureBox p = new PictureBox();

cockroach.Show(p, Field);

p.MouseMove += new MouseEventHandler(IMouseMove);

p.MouseDown += new MouseEventHandler(IMouseDown);

PB.Add(p);

LC.Add(cockroach);

workCockroach = cockroach;

workpb = p;

При загрузке формы должен создаваться первый Таракан.

Динамическое создание компонента PictureBox влечет динамическое создание обработчиков событий

p.MouseMove += new MouseEventHandler(IMouseMove);

p.MouseDown += new MouseEventHandler(IMouseDown);

Событие IMouseDown необходимо при выборе Таракана – исполнителя алгоритма

private void IMouseDown(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

int k = PB.IndexOf(sender as PictureBox);//запоминаем номер нажатого компонента PictureBox

workpb = sender as PictureBox;//объявляет его рабочим

workCockroach = LC[k];//по найденному номеру находим Таракана в списке

}

}

Событие IMouseMove необходимо при перетаскивании Таракана по полю

private void IMouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

PictureBox picture = sender as PictureBox;

picture.Tag = new Point(e.X, e.Y);//запоминаем координаты мыши на момент начала перетаскивания

picture.DoDragDrop(sender, DragDropEffects.Move);//начинаем перетаскивание ЧЕГО и с КАКИМ ЭФФЕКТОМ

}

}

Для компонента Field (панели) установите свойство AllowDrop в true и создайте 2 обработчика событий DragEnter и DragDrop.

В обработчике события DragEnter поместим

if (e.Data.GetDataPresent(typeof(PictureBox)))

{

e.Effect = DragDropEffects.Move;

}

Событие происходит при перетаскивании объекта в пределы границ элемента, выступающего в качестве объекта-приемника

В обработчике события DragDrop поместим

//извлекаем PictureBox

PictureBox picture = (PictureBox)e.Data.GetData(typeof(PictureBox));

Panel panel = sender as Panel;

//получаем клиентские координаты в момент отпускания кнопки

Point pointDrop = panel.PointToClient(new Point(e.X, e.Y));

//извлекаем клиентские координаты мыши в момент начала перетскивания

Point pointDrag = (Point)picture.Tag;

//вычисляем и устанавливаем Location для PictureBox в Panel

picture.Location = pointDrop;

//устанавливаем координаты для X и Y для рабочего таракана

workCockroach.X = picture.Location.X;

workCockroach.Y = picture.Location.Y;

picture.Parent = panel;

Событие DragDrop происходит при завершении операции перетаскивания

**Работа с алгоритмом для Таракана**

Для всех кнопок управления Тараканом в обработчике события Click должно быть добавление команды в список:

Algorithm.Items.Add((sender as Button).Text);

Очистку алгоритма напишите самостоятельно.

Запуск алгоритма на выполнение.

Алгоритм будет выполняться по командам по времени. Поэтому нужно при нажатии на кнопку Run запустить таймер и обнулить переменную AlgStep , отвечающую за номер команды в алгоритме. Эту переменную опишите в разделе описания полей формы.

Работа таймера:

if (AlgStep == Algorithm.Items.Count) //конец алгоритма

{

//выключение таймера

}

else//выполнение команды из списка

{

string s = (string)Algorithm.Items[AlgStep];

Algorithm.SetSelected(AlgStep, true);

if (s == "Step")

workCockroach.Step();

else

workCockroach.ChangeTrend(s[0]);

workCockroach.RePaint(workpb);

AlgStep++;

}

**Самостоятельная работа**

1. Выполните рефакторинг кода программы. Методы RePaint(PictureBox p) и Show(PictureBox p, Panel owner) не нужны в классе Cockroach. Перенесите логику отображения Таракана в класс Приложения (формы).
2. Реализовать множественный выбор исполнителей алгоритма при нажатой кнопке Ctrl. Выбранные исполнители смогут одновременно выполнять один и тот же алгоритм.
3. Добавьте возможность смены образа исполнителю.
4. Создать кнопку удаления таракана.