שיעורי בית יסודות מחלקת MTP – אופיר הופמן י3

```
class MTP
    {
        public enum Direction {up, upright, right, downright, down, downleft, left, upleft};
        private int x;
        private int y;
        private char ch;
        private ConsoleColor fcolor;
        private Direction direction;
        private int speed;
        private ConsoleColor bcolor;
        int MAX_X = 78;
        int MAX_Y = 23;
        Random rnd = new Random();

        public MTP(int x, int y, char ch, ConsoleColor fcolor, ConsoleColor bcolor, int speed)
        {
            this.x = x;
            this.y = y;
            this.ch = ch;
            this.fcolor = fcolor;
            this.bcolor = bcolor;
            this.speed = speed;
            this.direction = Direction.up;
        }

        public void Draw()
        {
            Console.SetCursorPosition(this.x, this.y);
            Console.ForegroundColor = this.fcolor;
            Console.Write(this.ch);
        }

        public void UnDraw()
        {
            Console.SetCursorPosition(this.x, this.y);
            Console.BackgroundColor = ConsoleColor.Black;
            Console.Write(" ");
        }




        public void MoveUp()
        {
            if ((this.y - this.speed) >= 0)
                this.y -= this.speed;
        }

        public void MoveDown()
        {
            if ((this.y + this.speed) <= 24)
                this.y += this.speed;
        }

        public void MoveRight()
        {
            if ((this.x + this.speed) <= 79)
                this.x += this.speed;
        }

        public void MoveLeft()
        {
            if ((this.x - this.speed) >= 0)
                this.x -= this.speed;
        }
```

```
public void MoveUpRight()
{
    if (((this.x + this.speed) <= 79) && (this.y - this.speed) <= 0)
    {
        this.x += this.speed;
        this.y -= this.speed;
    }
}

public void MoveUpLeft()
{
    if (((this.x - this.speed) >= 0) && ((this.y - this.speed) >= 0))
    {
        this.x -= this.speed;
        this.y -= this.speed;
    }
}

public void MoveDownRight()
{
    if (((this.x + this.speed) <= 79) && (this.y + this.speed) <= 24)
    {
        this.x += this.speed;
        this.y += this.speed;
    }
}

public void MoveDownLeft()
{
    if (((this.x - this.speed) >= 0) && (this.y + this.speed) <= 24)
    {
        this.x -= this.speed;
        this.y += this.speed;
    }
}

public void MoveOneStep()
{
    if ((int)direction == 0)
        MoveUp();
    if ((int)direction == 1)
        MoveUpRight();
    if ((int)direction == 2)
        MoveRight();
    if ((int)direction == 3)
        MoveDownRight();
    if ((int)direction == 4)
        MoveDown();
    if ((int)direction == 5)
        MoveDownLeft();
    if ((int)direction == 6)
        MoveLeft();
    if ((int)direction == 7)
        MoveUpLeft();
}



public void ChangeDirection()
{
    this.direction = (Direction)(((int)this.direction + 4) % 8);
}

public void HitBorederes()
{
    if (this.x > MAX_X || this.y > MAX_Y)
    {
        ChangeDirection();
    }
```

```
}

public bool Touch(MTP other)
{
    if (this.x == other.x && this.y == other.y)
    {
        return true;
    }
    return false;
}


public Direction GetDirection()
{
    return this.direction;
}

public void RndMove()
{
    int rndNum = rnd.Next(1, 11);
    if (rndNum == 5)
    {
        int rndDirection = rnd.Next(0, 8);
        this.direction = (Direction)rndDirection;
        MoveOneStep();
    }
}




public int GetX()
{
    return this.x;
}

public void SetX(int x)
{
    this.x = x;
}

public int GetY()
{
    return this.y;
}

public void SetY(int y)
{
    this.y = y;
}

public char getCh()
{
    return this.ch;
}

public void SetCh(char ch)
{
    this.ch = ch;
}

public ConsoleColor GetFcolor()
{
    return this.fcolor;
}

public void SetFcolor(ConsoleColor fcolor)
{
    this.fcolor = fcolor;
```

```
        }

        public int GetSpeed()
        {
            return this.speed;
        }

        public void SetSpeed(int speed)
        {
            this.speed = speed;
        }

        public ConsoleColor GetBcolor()
        {
            return this.bcolor;
        }

        public void SetBcolor(ConsoleColor bcolor)
        {
            this.bcolor = bcolor;
        }
    }

internal class Program
    {
        public static void MoveMtp(MTP mtp)
        {
            mtp.UnDraw();
            mtp.RndMove();
            mtp.Draw();
        }

        public static void TouchMtp3(MTP mtp, MTP other)
        {
            if(mtp.Touch(other))
            {
                Console.Beep();
            }
        }


        static void Main(string[] args)
        {

            MTP mtp1 = new MTP(1 ,1, '#', ConsoleColor.Blue, ConsoleColor.Red, 1);
            MTP mtp2 = new MTP(10, 18, '$', ConsoleColor.Red, ConsoleColor.Blue, 2);
            MTP mtp3 = new MTP(44, 25, '@', ConsoleColor.Green, ConsoleColor.White, 1);
            MTP mtp4 = new MTP(44, 25, '*', ConsoleColor.Magenta, ConsoleColor.Yellow, 3);
            MTP mtp5 = new MTP(30, 35, '%', ConsoleColor.Cyan, ConsoleColor.DarkRed, 2);

            TouchMtp3(mtp3, mtp4);

            bool cont = true;

            while(cont)
            {
                mtp1.HitBorederS();
                MoveMtp(mtp1);
                mtp2.HitBorederS();
                MoveMtp(mtp2);
                mtp3.HitBorederS();
                if (Console.KeyAvailable)
                {
                    mtp3.UnDraw();
                    ConsoleKeyInfo k = Console.ReadKey();
                    if (k.Key == ConsoleKey.UpArrow)
                        mtp3.MoveUp();
                    else if (k.Key == ConsoleKey.DownArrow)
                        mtp3.MoveDown();
```

```
                else if (k.Key == ConsoleKey.LeftArrow)
                    mtp3.MoveLeft();
                else if (k.Key == ConsoleKey.RightArrow)
                    mtp3.MoveRight();
                else if (k.Key == ConsoleKey.Escape)
                    cont = false;
                mtp3.Draw();
            }
            Thread.Sleep(50);
        }

    }
}
```