федеральное ГОСУДАРСТВЕННОЕ Бюджетное ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

**"САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ "ЛЭТИ" им.В.И.Ульянова (Ленина)" (СПбГЭТУ)**

Кафедра МОЭВМ

**Отчет по лабораторной работе №2**

«Основные проектные решения»

Выполнил: Морозов А.В.

Факультет КТИ

Группа № 0304

Преподаватель: Смольянинов А.В.

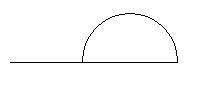
Санкт-Петербург

2013

## Формулирование основной идеи построения программы

Основным объектом задания является схематичный набор изображений «бейсболка мелкая», где каждое изображение образовано из следующих частей:

1. «Козырек» – представляется “линией”, задаётся с помощью двух точек;
2. «Шапка» – представляется ”сегментом круга ”, задаётся с помощью двух точек и угла разворота (центр и точка, относительно которой разворачивается дуга сегмента).

Набор изображений «Бейсболок» должен быть представлен в виде списка, разработанного в работе №1.  
 

## Построение системы объектов

Для построения объекта «Шапка» определим класс «Сегмент круга», а для объекта «Козырек» – класс «Линия».

### Объект «Точка»

|  |  |
| --- | --- |
| Декартовы координаты точки на плоскости. | float x, y; |
| Создаёт новую точку с координатами . | Point(); |
| Создаёт новую точку с координатами в прямоугольной системе координат. | Point(float x, float y); |
| Создаёт копию точки p. | Point(const Point &p); |
| Удаляет экземпляр точки. | ~Point(); |
| Создает точку, повернутую от данной относительно нуля на угол angle. | Point Rotate(float angle); |
| Создает точку, повернутую от данной относительно указанной точки center на угол angle. | Point RotateAround(float angle, Point center); |
| Создает точку, сдвинутую относительно данной на указанные сдвиги. | Point MoveBy(Point dxdy); |

### Объект «Линия»

|  |  |
| --- | --- |
| Начало и конец линии. | Point start, end; |
| Создает новую линию из начала и конца. | Line(Point start, Point end); |
| Создает копию линии. | Line(const Line &obj); |
| Удаляет экземпляр линии. | ~Line(); |
| Возвращает начало линии. | Point GetStart(); |
| Возвращает конец линии. | Point GetEnd(); |
| Двигает линию на указанный сдвиг. | void MoveBy(Point dxdy); |
| Вращает конец линии относительно начала. | void RotateEnd(float radians); |

### Объект «Сегмент круга»

|  |  |
| --- | --- |
| Центр и начало сегмента. | Point center, start; |
| Угол разворота сегмента. | float angle; |
| Создает новый сегмент круга из указанных центра, точки начала и угла разворота. | Segment(Point center, Point start, float angle); |
| Создает копию сегмента круга. | Segment(const Segment &obj); |
| Удаляет сегмент круга. | ~Segment(); |
| Возвращает центр сегмента. | Point GetCenter(); |
| Возвращает точку начала сегмента. | Point GetStart(); |
| Возвращает угол разворота сегмента. | float GetAngle(); |
| Двигает сегмент круга на указанный сдвиг. | void MoveBy(Point dxdy); |
| Двигает начало сегмента относительно его центра. | void RotateStart(float radians); |
| Устанавливает новый угол разворота. | void SetAngle(float angle); |

### Объект «Шапка»

|  |  |
| --- | --- |
| Квадрат, представляющий горелку. | Rectangle box; |
| Создаёт новую горелку с длиной стороны length и располагающуюся в position. | Burner(const Point &position, float length); |
| Создаёт копию горелки. | Burner(const Burner &b); |
| Удаляет экземпляр горелки. | ~Burner(); |
| Получает набор характерных точек. | void GetPoints(  CyclicList<Point \*> &list); |
| Получает длину боковой стороны. | float GetLength() const; |

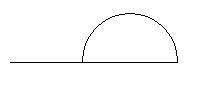
### Объект «Козырек»

|  |  |
| --- | --- |
| Трапеция, представляющая чашу. | Trapezium trap; |
| Создаёт новую чашу с высотой height и располагающуюся в position. | Bowl(const Point &position, float height); |
| Создаёт копию чаши. | Bowl(const Bowl &b); |
| Удаляет экземпляр чаши. | ~Bowl(); |
| Получает набор характерных точек. | void GetPoints(  CyclicList<Point \*> &list); |
| Получает высоту. | float GetHeight() const; |

### Объект «Кольцевой список» и «Элемент списка»

Были спроектированы и реализованы в лабораторной работе №1.

## Математические модели объектов

  
Для построения используется декартовая система координат.

P1, P2, P3, P4 – координаты горелки, чаши, корпуса и рукоятки соответственно

a – ширина горелки

h1

h2

s1, s2 – верхнее и нижнее основания чаши

s3, s4 – верхнее и нижнее основания корпуса

h1, h2, h3 – высота чаши, корпуса и рукоятки соответственно

w, h – ширина и высота рукоятки

С использованием указанных обозначений можно записать условия на местоположение и параметры фигур:

, при

, при

## Проектирование классов объектов на языке C++ на основе контрактов

В графах “Предусловия” и “Постусловия” и “Инвариант” предполагается, что значения типа данных float не содержат значений NaN, ±∞ и соответствуют множеству , а значения Point соответствуют множеству .

### Класс «Горелка»

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название | Предусловия | Постусловия |
| 1 | Burner(float length, Point position); | length > 0 | box.width = box.height = length  box.position = position + (-width / 2, height) |
| 2 | Burner(const Burner &p); | p – корректная горелка | Burner::box = p.box |
| 3 | ~Burner(); | - | экземпляр удалён |
| 4 | float GetLength() const; | - | результат = box.width =  = box.height > 0 |

**Инвариант**:

### Класс «Чаша»

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название | Предусловия | Постусловия |
| 1 | Bowl(float height, Point position); | height > 0 | trap.height = height  trap.side1 = 2 × height  trap.side2 = 1.5 × height  trap.position = position + (-side1 / 2, height) |
| 2 | Bowl(const Bowl &p); | p – корректная чаша | trap = p.trap |
| 3 | ~Bowl(); | - | экземпляр удалён |
| 4 | float GetHeight() const; | - | результат = trap.height > 0 |

**Инвариант**:

### Класс «Корпус»

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название | Предусловия | Постусловия |
| 1 | Body(float height, Point position); | height > 0 | trap.height = height  trap.side1 = 0.2 × height  trap.side2 = 0.1 × height  trap.position = position + (-side1 / 2, height) |
| 2 | Body(const Body &p); | p – корректный корпус | trap = p.trap |
| 3 | ~Body(); | - | экземпляр удалён |
| 4 | float GetHeight() const; | - | результат = trap.height > 0 |

**Инвариант**:

### Объект «Рукоятка»

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название | Предусловия | Постусловия |
| 1 | Handle(float width, Point position); | width > 0 | box.width = width  box.height = 1.5 × width  box.position = position + (-width / 2, height) |
| 2 | Handle(const Handle &p); | p – корректная рукоятка | box = p.box |
| 3 | ~Handle(); | - | экземпляр удалён |
| 4 | float GetWidth() const; | - | результат = box.width > 0 |

**Инвариант**:

### Объект «Факел»

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название | Предусловия | Постусловия |
| 1 | Torch(float height,  Point position); | height > 0 | burner.GetLength() = height  bowl.GetHeight() = height  body.GetHeight() = height  handle.GetWidth() = height  Torch::position = position |
| 2 | Torch(const Torch &p); | p – корректный факел | Torch::burner = p.burner  Torch::bowl = p.bowl  Torch::body = p.body  Torch::handle = p.handle  Torch::position = p.position |
| 3 | ~Torch(); | - | экземпляр удалён |
| 4 | const Burner & GetBurner() const; | - | результат – корректная горелка |
| 5 | const Trapezium & GetBowl() const; | - | результат – корректная чаша |
| 6 | const Trapezium & GetBody() const; | - | результат – корректный корпус |
| 7 | const Handle & GetHandle() const; | - | результат – корректная рукоятка |
| 8 | void MoveTo(Point p); | - | Torch::position = p |
| 9 | void MoveBy(float dx, float dy); | - | Torch::position’ = Torch::position + (dx, dy) |

**Инвариант**:

body.position = burner.position + (bowl.position − burner.position)

handle.position = burner.position + (bowl.position − burner.position)

&& distance(burner.position, bowl.position) = bowl.GetHeight() = 2 × burner.GetLength()

&& distance(bowl.position, body.position) = body.GetHeight() = 10 × burner.GetLength()

&& distance(body.position, handle.position) = handle.GetHeight() = 1.5 × burner.GetLength()

где distance(u, v) =

&& handle.GetWidth() = burner.GetLength()

### Объект «Кольцевой список» и «Элемент списка»

Описания приведены в отчёте к лабораторной работе №1.

В качестве критерия, по которому производится разделение набора фигур на поднаборы фигур, выступает положительность или неположительность y-координаты положения факела на плоскости.