МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ЛУГАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ ВЛАДИМИРА ДАЛЯ»

Факультет компьютерных систем и информационных технологий
Кафедра информационных и управляющих систем

УТВЕРЖДАЮ
Декан факультета компьютерных систем и информационных технологий
_____ Кочевский А.А.
«19» _____ 2023 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
по учебной дисциплине

«Объектно-ориентированное программирование»

44.03.04. Профессиональное обучение (по отраслям)

«Информатика и вычислительная техника»

Разработчик:
Старший преподаватель кафедры
информационных и управляющих систем _____ Балалаечников А.В.

ФОС рассмотрен и одобрен на заседании кафедры информационных и управляющих систем от «18» апреля 2023 г., протокол № 15.

Заведующий кафедрой
информационных и управляющих систем _____Горбунов А.И.

Луганск 2023 г.

Паспорт
фонда оценочных средств по учебной дисциплине
«Объектно-ориентированное программирование»

Перечень компетенций (элементов компетенций), формируемых в результате освоения учебной дисциплины

№ п/п	Код контролируемой компетенции	Формулировка контролируемой компетенции	Контролируемые темы учебной дисциплины	Этапы формирования (семестр изучения)
1	УК-1	Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	Тема №1. Основы языка Object Pascal	начальный (4)
			Тема №2. Введение в концепцию ООП	
			Тема №3. Визуальные компоненты VCL и ООП	
			Тема №4. Разработка отказоустойчивых приложений	
			Тема №5. Понятие объектно-ориентированного программирования	
			Тема №6. Классы и объекты общего назначения	
			Тема №7. Понятие наследования	
			Тема №8. Полиморфизм	
			Тема №9. Декомпозиция и абстракция	
2	ПК-4	Способен разрабатывать, обновлять программное и учебно-методическое обеспечение учебных предметов, курсов, дисциплин (модулей), практик и планировать занятия	Тема №1. Основы языка Object Pascal	начальный (4)
			Тема №2. Введение в концепцию ООП	
			Тема №3. Визуальные компоненты VCL и ООП	
			Тема №4. Разработка отказоустойчивых приложений	
			Тема №5. Понятие объектно-ориентированного программирования	
			Тема №6. Классы и объекты общего назначения	
			Тема №7. Понятие наследования	
			Тема №8. Полиморфизм	
			Тема №9. Декомпозиция и абстракция	
3	ПК-5	Способен организовывать контроль и оценку освоения образовательной программы профессионального обучения, СПО и (или) ДПП в процессе учебно-производственной	Тема №1. Основы языка Object Pascal	начальный (4)
			Тема №2. Введение в концепцию ООП	
			Тема №3. Визуальные компоненты VCL и ООП	
			Тема №4. Разработка отказоустойчивых приложений	
			Тема №5. Понятие объектно-ориентированного программирования	

№ п/п	Код контролируемой компетенции	Формулировка контролируемой компетенции	Контролируемые темы учебной дисциплины	Этапы формирования (семестр изучения)
		деятельности	Тема №6. Классы и объекты общего назначения	
			Тема №7. Понятие наследования	
			Тема №8. Полиморфизм	
			Тема №9. Декомпозиция и абстракция	
4	ПК-8	Способен выполнять работы (услуги), организовывать их выполнение и контроль их качества в соответствии с требованиями нормативной и технической документации и нормами времени на выполнение соответствующих работ (в зависимости от реализуемой образовательной программы, преподаваемого учебного предмета, курса, дисциплины (модуля))	Тема №1. Основы языка Object Pascal	начальный (4)
			Тема №2. Введение в концепцию ООП	
			Тема №3. Визуальные компоненты VCL и ООП	
			Тема №4. Разработка отказоустойчивых приложений	
			Тема №5. Понятие объектно-ориентированного программирования	
			Тема №6. Классы и объекты общего назначения	
			Тема №7. Понятие наследования	
			Тема №8. Полиморфизм	
			Тема №9. Декомпозиция и абстракция	

Показатели и критерии оценивания компетенций, описание шкал оценивания

№ п/п	Код контролируемой компетенции	Показатель оценивания (знания, умения, навыки)	Контролируемые темы учебной дисциплины	Наименование оценочного средства
1	УК-1	Знать основные источники и методы поиска информации, необходимой для решения поставленных задач, законы и формы логически правильного мышления, основы теории аргументации, сущность и основные принципы системного подхода Уметь осуществлять поиск информации для решения поставленных задач и критически ее анализировать; применять методы критического анализа и синтеза информации, необходимой для решения поставленных задач; применять законы логики и основы теории аргументации при осуществлении критического анализа и синтеза информации, необходимой для решения поставленных задач; грамотно, логично, аргументированно формировать собственные суждения и оценки; отличать факты от мнений,	Тема 1, Тема 2, Тема 3, Тема 4, Тема 5, Тема 6, Тема 7, Тема 8, Тема 9	Контрольные вопросы, практические работы

№ п/п	Код контролируемой компетенции	Показатель оценивания (знания, умения, навыки)	Контролируемые темы учебной дисциплины	Наименование оценочного средства
		интерпретаций и оценок; применять методы системного подхода при решении поставленных задач Владеть методами системного и критического мышления		
2	ПК-4	Знать требования ФГОС, содержание примерных (типовых) программ; требования профессиональных стандартов по соответствующему виду профессиональной деятельности; требования и методические основы при разработке программно-методического обеспечения учебных предметов, курсов, дисциплин (модулей), практик; современное состояние области знаний и (или) профессиональной деятельности, соответствующей преподаваемым учебным предметам, курсам, дисциплинам (модулям), практикам Уметь анализировать учебно-методическое обеспечение учебных предметов, курсов, дисциплин (модулей), практик Владеть навыками разработки и обновления программного и учебно-методического обеспечения учебных предметов, курсов, дисциплин (модулей), практик; заданий для самостоятельной работы	Тема 1, Тема 2, Тема 3, Тема 4, Тема 5, Тема 6, Тема 7, Тема 8, Тема 9	Контрольные вопросы, практические работы
3	ПК-5	Знать нормативные правовые акты, локальные нормативные акты организации, регламентирующие процедуры оценки освоения обучающимися образовательной программы профессионального обучения, СПО и (или) ДПП в процессе учебно-производственной деятельности; современные подходы, методы и инструментарий мониторинга и оценки качества реализации программ учебных предметов, курсов, дисциплин (модулей), практик, анализа занятий Уметь планировать и проводить мониторинг и оценку качества реализации программ учебных предметов, курсов, дисциплин (модулей), практик; планировать систему корректирующих и предупреждающих действий; анализировать состояние, планировать и организовывать методическую деятельность в образовательной организации Владеть приемами и методами контроля учебной и учебно-производственной деятельности обучающихся по освоению программ профессионального обучения и(или) программ подготовки квалифицированных рабочих, служащих; организовывать или проводить контроль и оценку учебной и(или) производственной практики (практического обучения)	Тема 1, Тема 2, Тема 3, Тема 4, Тема 5, Тема 6, Тема 7, Тема 8, Тема 9	Контрольные вопросы, практические работы
4	ПК-8	Знать современные отраслевые технологии, обеспечивающие решение профессиональных задач, выполнение	Тема 1, Тема 2,	Контрольные вопросы,

№ п/п	Код контролируемой компетенции	Показатель оценивания (знания, умения, навыки)	Контролируемые темы учебной дисциплины	Наименование оценочного средства
		трудовых функций, технологических операций (по профилю подготовки) Уметь использовать современные отраслевые технологии, обеспечивающие решение профессиональных задач, выполнение трудовых функций, технологических операций (по профилю подготовки) в процессе профессионального обучения по программам СПО и (или) ДПП Владеть навыками использования современных отраслевых технологий, обеспечивающие решение профессиональных задач, выполняющие трудовые функции, технологические операции (по профилю подготовки) в процессе профессионального обучения по программам СПО и (или) ДПП	Тема 3, Тема 4, Тема 5, Тема 6, Тема 7, Тема 8, Тема 9	практические работы

Фонды оценочных средств по дисциплине «Объектно-ориентированное программирование»

Пример типовой практической работы №1.

ПРАКТИЧЕСКАЯ РАБОТА №1. «ИСПОЛЬЗОВАНИЕ БАЗОВЫХ ОПЕРАТОРОВ ДЛЯ ВЫЧИСЛЕНИЯ ФОРМУЛ»

Цель: Вычислить значение функции, заданной по условию. Организовать ввод значений и вывод результата вычисления.

1.1 Теоретические сведения

1.1.1 Правила записи стандартных функций

1. Имя функции записывается латинскими буквами.
2. Аргумент функции записывается в круглых скобках после имени функции.
3. Аргументом функции может быть константа, переменная или выражение соответствующего типа.

ЗАМЕЧАНИЯ.

В тригонометрических функциях синус и косинус аргумент может быть задан только в радианной мере.

1.1.2 Арифметические функции

Функция	Назначение	Тип аргумента	Тип функции
Abs(x)	|x|	Real Integer	Real Integer
Sqr(x)	a^2	Real Integer	Real Integer
Sqrt(x)	$\sqrt{x}$	Real Integer	Real Real
sin(x)	sin x	Real Integer	Real Real
cos(x)	cos x	Real Integer	Real Real
tan(x)	tg x	Real Integer	Real Real
arctan(x)	arctg x	Real Integer	Real Real

exp(x)	e^x	Real Integer	Real Real
ln(x)	ln x	Real Integer	Real Real
int(x)	[x]	Real Integer	Real Real
Random(x)	Возвращает случайное целое неотрицательное число <x	Word	Word
Random	Возвращает случайное вещественное неотрицательное число<1		Real
Power(x,y)	Возводит X в степень Y. Перед использованием подключить в uses модуль math	Real, Real	Real

1.2 Ход выполнения работы

```
program Project1;
{$APPTYPE CONSOLE}
//описываем подключаемые модули
// модуль SysUtils - стандартная библиотека
// модуль math - библиотека матем. функций
uses
  SysUtils, math;
var
    y,x,d,f: real;
// Описываем переменные для вычисления формул
begin
//Вводим значения переменных d, f
writeln('Vvedite D');
readln(d);
writeln('Vvedite F');
readln(f);
//вычисляем X
x:= 8 - ln(d*d -f);
//вычисляем Y
y:= sqrt(cos(x)*cos(x) + sin(x-2)/power(x,3));
//выводим результаты вычисления
writeln('Znachenie x=', x);
writeln('Znachenie y(x)=', y);
readln;
end.
```

ПРАКТИЧЕСКАЯ РАБОТА №2. «РАЗРАБОТКА И РЕАЛИЗАЦИЯ КЛАССА ДВУМЕРНЫХ ПРИМИТИВОВ»

Цель: Реализовать класс примитива «Круг», который имеет координаты в декартовой системе, радиус, цвет, может перемещаться изменяя координаты согласно приращений по осям X, У, может менять свой цвет, рисовать себя на холсте (Canvas). Визуально продемонстрировать работу с объектами данного класса.

2.1 Ход выполнения работы

Создадим новый проект в среде Delphi. **File-New VCL Form Application**

Перейдем в окно редактора, и выше интерфейса класса нашей формы опишем интерфейс нашего класса. Который будет иметь следующий вид:

```
//Класс "Круг"
TCircle = class
private
  //Илентификатор круга. Для того чтобы различать объекты данного класса между собой
  ID: integer;
  //Радиус круга
  Radius: integer;
  //Координаты центра круга
  X, Y: integer;
  //Приращение перемещения круга по осям X,Y
  dX, dY :integer;
  // Цвет круга
  Color: TColor;
  // Холст на котором рисуем круг
  Canvas: TCanvas;
  // состояние движения круга. true - движется, false - остановлен
  InMotion: boolean;
public
  //Метод отвечающий за рисование круга на текущем холсте
  procedure Draw;
  //Метод отвечающий за движение круга. Меняет координа на приращение
  procedure Move;
  //Метод останавливает движение круга
  procedure Stop;
  //Свойство для доступа к состоянию движения. Позволяет менять его
  property Motion: boolean read InMotion write InMotion;
  //Свойство для доступа к координатам объекта
  property circle_X: integer read x write x;
  property circle_Y: integer read y write y;
  //Метод для смены цвета круга
  procedure ChangeColor (vColor: TColor);
  //Метод вычисления площади
  function Area: real;
  //Конструктор создания объекта. Принимает переменные для инициализации полей объета класса.
  constructor Create (vID, vX,vY,vRadius: integer; vColor: TColor; vCanvas: TCanvas);
end;
```

Поля класса размещаем в секции **private** для ограничения доступа к ним, реализуем доступ посредством **свойств** к координатам X,Y и состоянию движения inMotion. Ко всем остальным полям доступ не предоставляем. В дальнейшем можно будет дописать свойства для доступа к остальным данным. Методы и свойства размещаем в разделе **public –** предоставляем общий доступ к ним.

Кроме того добавим к нашей форме 2 поля которые будут отвечать за хранение объектов нашего класса.

Для этого **в разделе private формы** допишем код.

```
private
  ArrCircles: array [1..30] of TCircle;
  CountCircles: integer;
```

Массив из 30 элементов тип данных у которых представляет наш класс, описанный выше, и переменная для хранения текущего количества созданных объектов класса **TCircle.**

Далее опишем 2 глобальных переменных в которых будут храниться размеры окна формы. В дальнейшем они нам понадобятся для рисования движущихся кругов.

```
var
  MainForm: TMainForm;
  HeightCanvas: integer;
  WidthCanvas: integer;
```

После того как интерфейс класса был описан, достаточно нажать комбинацию **Ctrl+Shift+C** чтобы создались каркасы методов класса в разделе реализации. Опишем их.

Метод вычисляет площадь круга и возвращает ее значение.

```
function TCircle.Area: real;
begin
  result := 3.14159 * sqr(radius);
end;
```

Конструктор, в который передаются значения для инициализации полей объекта. Координаты, радиус, цвет, холст и номер круга в массиве. Они присваиваются полям,

скорость перемещения мы вычисляем сами внутри конструктора случайным образом. Наш создаваемый круг будет иметь хаотическое движение в случайном направлении.

```
constructor TCircle.Create(vID, vx, vy, vradius: integer; vColor: TColor;
  vCanvas: TCanvas);
begin
  Randomize;
  ID := vID;
  x := vx;
  y := vy;
  //Скорость перемещения по X и Y берем случайным образом.
  dx := 20-Random(20);
  dy := 20-Random(20);
  if dx = 0 then dx := 8;
  if dy = 0 then dy := 7;
  radius := vradius;
  Color := vColor;
  Canvas := vCanvas;
  //Объект по умолчанию создается сразу в движении
  InMotion := true;
end;
```

Метод рисования объекта изображает его на холсте, который мы передаем в конструкторе. Рисуем круг заданного цвета с номером по центру, чтобы различать объекты между собой.

```
procedure TCircle.Draw;
begin
  with Canvas do
  begin
    //Устанавливаем кисть и перо для рисования круга
    pen.Color := clwhite;
    pen.Width := 1;
    Brush.Color := Color;
    //Рисуем круг с заданными координатами и радиусом
    Ellipse(rect(x-Radius,y-Radius,x+Radius,y+Radius));
    font.Color := clWhite;
    font.Style := [fsBold];
    //Выводим ID объекта внутри круга
    if ID < 10 then
      TextOut(x-Radius+8,y-Radius+6,inttostr(ID))
    else
      TextOut(x-Radius+4,y-Radius+5,inttostr(ID))
  end;
end;
```

Метод отвечающий за перемещение круга по холсту, изменяет координаты до тех пор, пока объект не дойдет до границ, затем меняет приращение, эмулируя отражение от границы.

```
procedure TCircle.Move;
begin
  //Если объект в движении
  if InMotion then
  begin
    //Если объект выходит за пределы размеров формы, меняем приращение на обратное
    //И перемещаем объект в новые координаты, меняем X,Y
    if (x+radius > WidthCanvas) or (x-radius < 0) then dx := -dx;
    x := x + dx;
    if (y+radius > HeightCanvas) or (y-radius < 0) then dy := -dy;
    y := y + dy;
  end;
end;
```

Метод отвечает за остановку круга.

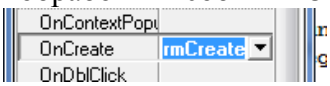
```
procedure TCircle.Stop;
begin
  //Останавливаем объект
  InMotion := false;
end;
```

Метод отвечает за изменение цвета круга.

```
procedure TCircle.ChangeColor(vColor: TColor);
begin
  Color := vColor;
end;
```

На данном этапе у нас есть законченный класс с реализацией, который описывает некое поведение круга в декартовой системе координат. Теперь используем этот класс чтобы визуально увидеть как это работает на практике.

В **Object Inspector** создадим обработчик события **onCreate** для нашей формы:



С следующим кодом:

```
procedure TMainForm.FormCreate(Sender: TObject);
begin
  //Включаем буфер, для плавной перерисовки формы
  DoubleBuffered := True;
  //Начальное количество кругов
  CountCircles := 0;
  //Вычисляем размеры Canvas окна формы
  HeightCanvas := Canvas.ClipRect.Bottom;
  WidthCanvas := Canvas.ClipRect.Right;
end;
```

Вычисляем размеры формы, заносим их в переменные. Устанавливаем количество кругов в начальное значение.

В **Object Inspector** создадим обработчик события **onDestroy** для нашей формы с следующим кодом:

```
procedure TMainForm.FormDestroy(Sender: TObject);
var i: integer;
begin
  // проходим по массиву кругов и удаляем их
  for i:=1 to CountCircles do ArrCircles[i].Free;
end;
```

При закрытии формы в цикле для всех объектов класса **TCircle** вызываем деструктор.

В **Object Inspector** создадим обработчик события **onResize** для нашей формы с следующим кодом:

```
procedure TMainForm.FormResize(Sender: TObject);
var i:integer;
begin
  //Вычисляем размеры Canvas окна формы
  HeightCanvas := Canvas.ClipRect.Bottom;
  WidthCanvas := Canvas.ClipRect.Right;
  for i:=1 to CountCircles do
  begin
    ArrCircles[i].circle_X := WidthCanvas div 2;
    ArrCircles[i].circle_Y := HeightCanvas div 2;
    ArrCircles[i].Move;
    ArrCircles[i].Motion := true;
  end;
end;
```

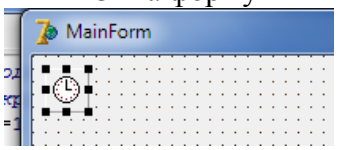
При изменении размера формы, пересчитываем размеры для просчета движения кругов, изменяем им позицию в центр формы и запускаем их в движение из этой точки.

В **Object Inspector** создадим обработчик события **onPaint** для нашей формы с следующим кодом:

```
procedure TMainForm.FormPaint(Sender: TObject);
var i: integer;
begin
  MainForm.Canvas.Brush.color:=clWhite;
  MainForm.Canvas.FillRect(MainForm.Canvas.ClipRect);
  // проходим по массиву arrCircles
  for i:=1 to CountCircles do begin
  // перемещаем круг
   ArrCircles[i].Move;
   // рисуем круг
   ArrCircles[i].Draw;
  end;
end;
```

Это метод отвечает за рисование кругов на форме. Метод будет вызываться автоматически при каждой перерисовке формы. Будет по циклу заливать форму белым цветом, у кругов вызывать метод движения для изменения координат и метод рисования на холсте, переданном в объект.

Далее разместим компонент **TTimer** на форму



Установим ему имя и значение интервала в миллисекундах

И создадим ему обработчик события **onTimer**

```
procedure TMainForm.DrawTimerTimer(Sender: TObject);
begin
  //По таймеру перерисовываем форму. Автоматом будет вызываться метод формы onPaint
  MainForm.Repaint;
end;
```

Он будет срабатывать при каждом тике компонента с интервалом. Будет вызывать перерисовку формы, в которой задано изменение координат кругов и так до бесконечности.

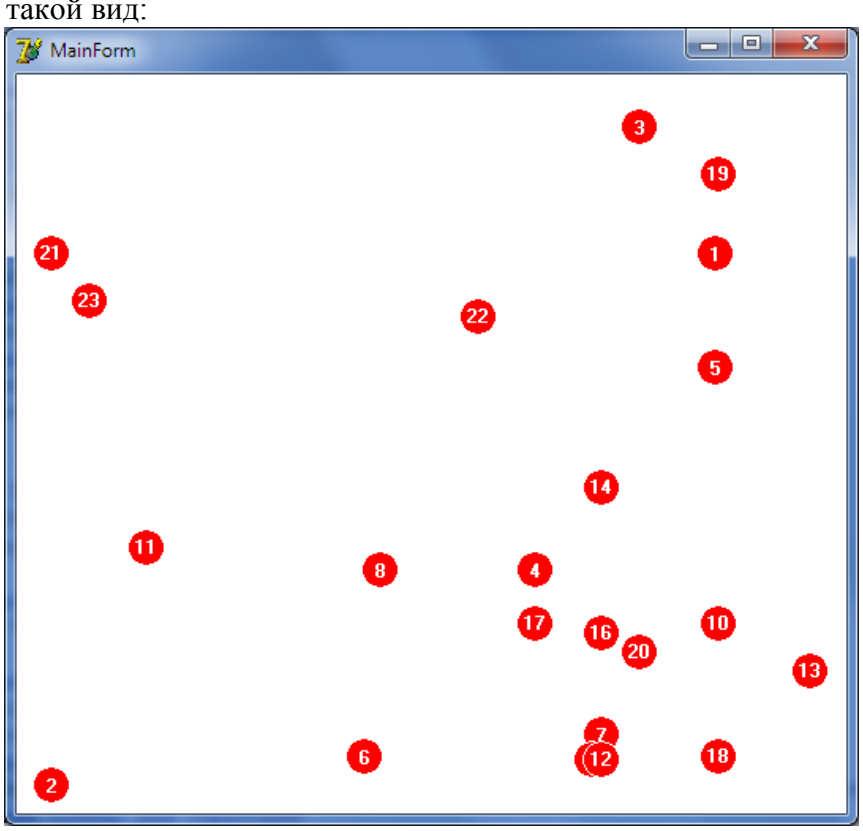
Теперь перейдем к последнему этапу, созданию объектов класса в памяти. Для этого создадим обработчик события **onMouseDown** для формы.

```
procedure TMainForm.FormMouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
var i:integer;
begin
//Если нажата левая кнопка мыши
if Button = mbLeft then
begin
// Проверяем если количество примитивов меньше 30 то увеличиваем на +1
if CountCircles = 30 then exit;
  CountCircles := CountCircles + 1;
  // Создаем объект класса TCircle и размещаем в массиве примитивов
  ArrCircles [CountCircles]:=TCircle.Create(CountCircles,x,y,12,clRed,MainForm.Canvas);
  //Включаем таймер для отрисовки
  DrawTimer.Enabled:=true;
end
//Если нажата правая кнопка мыши
else if Button = mbRight then
begin
  //Проходим по массиву и меняем свойство отвечающее за движение на обратное.
  // Те круги, которые двигались остановятся, те которые были остановлены ранее, начнут движение.
  for i:=1 to CountCircles do ArrCircles[i].Motion := not (ArrCircles[i].Motion);
end;
end;
```

По нажатию левой кнопки мыши, в заданных координатах курсора будет создаваться объект **TCircle** красного цвета с радиусом 12 пикселов и начинать «произвольное

движение» по форме. По нажатию правой кнопки мыши все созданные объекты меняют свое состояние движения на обратное.

Откомпилируйте исходный код и запустите на выполнение. Покликайте по форме левой и правой кнопкой мышки, измените размер формы. Результат работы приложения должен иметь такой вид:



Вопросы для защиты практических работ:

1. Лексика языка Паскаль (алфавит, имена, константы, знаки операций и т.д.)
2. Общая структура программы на языке Паскаль.
3. Типы данных. Базовые (скалярные) типы данных
4. Типы данных. Действительные типы данных.
5. Типы пользователя (счетно и интервальный (диапазон)) в языке Паскаль.
6. Простые операторы языка Паскаль (присвоение, перехода, вызова процедуры), составленный оператор.
7. типы данных. тем данных boolean (множество значений, множество операций, примеры использования).
8. математические операции в языке Pascal. приоритет выполнения операций.
9. Ввод и вывод информации в языке Паскаль.
10. Форматированный вывод в языке Паскаль.
11. условный оператор.
12. оператор выбора.
13. Циклы. цикл с предпосылкой.
14. Циклы. цикл с постусловием.
15. Циклы. цикл с параметром.
16. Одномерные массивы на языке Pascal
17. двумерные массивы языке Pascal.
18. Строчный тип данных (string) в языке Паскаль. Стандартные процедуры и функции для работы с строками.
19. Процедуры в языке Паскаль.
20. Функции в языке Паскаль.

21. Рекурсия. принцип выполнения рекурсивных подпрограмм. Примеры.
22. параметры процедур и функций в языке Паскаль.
23. Множества в языке Паскаль. операции над множествами.
24. Записи в языке Паскаль. оператор with.
25. Работа с файлами в языке Pascal.
26. Понятие о статических и динамических переменных.
27. Указатели в языке Паскаль.
28. особенности работы с динамическими переменными (утраченные ссылки и т.д.)
29. Общие сведения о файлах. Процедуры работы с файлами в языке Паскаль.
30. динамические структуры: списки. Основные операции с однонаправленных списками на примере языка Паскаль.
31. динамические структуры: дерева. Основные операции с бинарными деревьями на примере языка Паскаль.
32. Динамические структуры: стек. Основные операции с стеком на примере языка Паскаль.
33. динамические структуры: очередь. Основные операции с очередь на примере языка Паскаль.
34. Обработка исключительных ситуаций. Блок try … exept … end.
35. Обработка исключительных ситуаций. блок try … finally … end.
36. Модули в Паскаль Создание собственного модуля пользователя.
37. Понятие объекта в языке Pascal.
38. Объектные элементы синтаксиса языка программирования Object Pascal: классы, методы, свойства, события и реакции на них.
39. Объявление переменных и методов класса.
40. принципы объектного программирования.
41. Графические возможности языка программирования Object Pascal. понятие канвы.
42. Графические возможности языка программирования Object Pascal. Рисование прямых, ломаных и кривых линий.
43. Графические возможности языка программирования Object Pascal. рисование геометрических фигур.
44. Графические возможности языка программирования Object Pascal. заполнение внутреннего пространства фигур.
45. Перечислите классические типы. Опишите механизмы для создания новых типов.
46. В чем различие между видами и методами (способами) абстракции?
47. Дайте характеристику парадигме ООП и специфике интерфейса ОО-программ.
48. Дайте понятие класса в ООП. Опишите отношение "объект - класс".
49. Опишите механизм наследования.
50. В чем смысл декомпозиции при составлении программ?
51. Основные понятия ООП.
52. Программирование для Windows.
53. Технология разработки Windows-приложений в Delphi. Визуальное программирование.
54. Основные элементы среды разработки приложений Delphi.
55. Основные визуальные компоненты Delphi. Краткая характеристика.
56. Object Pascal. Вызов функций, передача параметров, возврат результатов, перегрузка функций.
57. Динамическое распределение памяти. Указатели, операции с указателями.
58. Динамические массивы.
59. Самоадресуемые записи, динамические списки.
60. Обработка исключительных ситуаций.
61. Понятие класса, объекта и компонента.
62. Составляющие класса. Объявление класса.

63. Поля и свойства классов.
64. Методы, конструкторы и деструкторы классов.
65. События и обработчики событий классов.
66. Механизм наследования и иерархия классов.
67. Алгоритм построения дерева иерархии классов.
68. Виртуальные методы и полиморфизм.
69. Замещение (перекрытие и перегрузка) методов.
70. Абстрактные методы и классы.
71. Операции с классами, ссылки на классы и информация о классах.
72. Регистрация и обнаружение классов.
73. Создание компонентов во время выполнения, клонирование объектов.
74. Поиск компонентов и уничтожение объектов во время выполнения программы.
75. Двукратное освобождение памяти.
76. Построение собственных компонентов – причины и требования к компонентам.
77. Этапы построения собственных компонентов.
78. Пиктограммы компонентов. Правила именования свойств, полей, методов, модулей, пиктограмм.
79. Типы компонентов Delphi. Фундаментальные классы компонентов библиотеки VCL.
80. Алгоритм построения дерева визуальных объектов формы.

Критерии и шкала оценивания по оценочному средству защита практических работ

Шкала оценивания (интервал баллов)	**Критерий оценивания**
отлично (5)	Студент глубоко и в полном объёме владеет программным материалом. Грамотно, исчерпывающе и логично его излагает в устной или письменной форме. При этом знает рекомендованную литературу, проявляет творческий подход в ответах на вопросы и правильно обосновывает принятые решения, хорошо владеет умениями и навыками при выполнении практических задач.
хорошо (4)	Студент знает программный материал, грамотно и по сути излагает его в устной или письменной форме, допуская незначительные неточности в утверждениях, трактовках, определениях и категориях или незначительное количество ошибок. При этом владеет необходимыми умениями и навыками при выполнении практических задач.
удовлетворительно (3)	Студент знает только основной программный материал, допускает неточности, недостаточно чёткие формулировки, непоследовательность в ответах, излагаемых в устной или письменной форме. При этом недостаточно владеет умениями и навыками при выполнении практических задач. Допускает до 30% ошибок в излагаемых ответах.
неудовлетворительно (2)	Студент не знает значительной части программного материала. При этом допускает принципиальные ошибки в доказательствах, в трактовке понятий и категорий, проявляет низкую культуру знаний, не владеет основными умениями и навыками при выполнении практических задач. Студент отказывается от ответов на дополнительные вопросы

Оценочные средства для промежуточной аттестации (экзамен)

1. Этапы решения задач с использованием ЭВМ.
2. Понятие алгоритма. Подходы к определению алгоритма. Свойства алгоритма. Способы записи алгоритма.
3. Понятие алгоритма. Понятие исполнителя. Система команд исполнителя.
4. Понятие величины. Типы величин. Присваивание величин. Совместимость по присваиванию.
5. Понятие о структурном программировании. Другие парадигмы программирования: сравнительная характеристика.
6. Языки программирования. Алгоритмические языки (алфавит, синтаксис, семантика). Способы описания синтаксиса (язык металингвистических формул, синтаксические диаграммы).
7. Система программирования Delphi.
8. Структура программы, элементы языка (алфавит). Понятие типа данных.
9. Операции (арифметические, логические) на типах. Стандартные функции. Выражения.
10. Процедуры консольного ввода и вывода, управление вводом-выводом. Оператор присваивания. Совместимость по присваиванию.
11. Условный оператор. Оператор множественного ветвления (выбора).
12. Циклы в Delphi: с предусловием, с постусловием. Связь с другими циклами.
13. Циклы в Delphi: с параметром. Связь с другими циклами.
14. Структурированные типы данных. Линейные массивы.
15. Структурированные типы данных. Двумерные массивы.
16. Подпрограммы в Delphi. Основные способы передачи параметров в подпрограмму, их сравнение.
17. Подпрограммы в Delphi. Область видимости. Локальные и глобальные идентификаторы.
18. Процедуры. Организация и вызов.
19. Функции. Организация и вызов.
20. Простые типы данных в Delphi.
21. Структурированные типы данных. Строковый тип данных в Delphi: основные процедуры и функции, примеры.
22. Основные понятия ООП.
23. Программирование для Windows.
24. Технология разработки Windows-приложений в Delphi. Визуальное программирование.
25. Основные элементы среды разработки приложений Delphi.
26. Основные визуальные компоненты Delphi. Краткая характеристика.
27. Object Pascal. Вызов функций, передача параметров, возврат результатов, перегрузка функций.
28. Динамическое распределение памяти.
29. Указатели, операции с указателями.
30. Динамические массивы.
31. Обработка исключительных ситуаций.
32. Понятие класса, объекта и компонента.
33. Составляющие класса. Объявление класса.
34. Поля и свойства классов.
35. Методы, конструкторы и деструкторы классов.
36. События и обработчики событий классов.
37. Механизм наследования и иерархия классов.
38. Алгоритм построения дерева иерархии классов.
39. Виртуальные методы и полиморфизм.

40. Замещение (перекрытие и перегрузка) методов.
41. Абстрактные методы и классы.
42. Операции с классами, ссылки на классы и информация о классах.
43. Регистрация и обнаружение классов.
44. Создание компонентов во время выполнения, клонирование объектов.
45. Поиск компонентов и уничтожение объектов во время выполнения программы.
46. Двукратное освобождение памяти.
47. Построение собственных компонентов – причины и требования к компонентам.
48. Этапы построения собственных компонентов.
49. Пиктограммы компонентов. Правила именования свойств, полей, методов, модулей, пиктограмм.
50. Типы компонентов Delphi. Фундаментальные классы компонентов библиотеки VCL.

Типовой экзаменационный билет.

ЛУГАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. В.ДАЛЯ

Кафедра информационных и управляющих систем

Вариант 1

1. Составляющие класса. Объявление класса.
2. Методы, конструкторы и деструкторы классов.
3. Абстрактные методы и классы.

Утверждено на заседании кафедры информационных и управляющих систем _____, протокол_____

Заведующий кафедрой А.И. Горбунов

Экзаменатор А.В. Балалаечников

Критерии и шкала оценивания по оценочному средству промежуточный контроль (экзамен)

Шкала оценивания (интервал баллов)	**Критерий оценивания**
отлично (5)	Студент глубоко и в полном объёме владеет программным материалом. Грамотно, исчерпывающе и логично его излагает в устной или письменной форме. При этом знает рекомендованную литературу, проявляет творческий подход в ответах на вопросы и правильно обосновывает принятые решения, хорошо владеет умениями и навыками при выполнении практических задач.
хорошо (4)	Студент знает программный материал, грамотно и по сути излагает его в устной или письменной форме, допуская незначительные неточности в утверждениях, трактовках, определениях и категориях или незначительное количество ошибок. При этом владеет необходимыми умениями и навыками при выполнении практических задач.
удовлетворительно (3)	Студент знает только основной программный материал, допускает неточности, недостаточно чёткие формулировки, непоследовательность в ответах, излагаемых в устной или письменной форме. При этом недостаточно владеет умениями и навыками при выполнении практических задач. Допускает до 30% ошибок в излагаемых ответах.
неудовлетворительно (2)	Студент не знает значительной части программного материала. При этом допускает принципиальные ошибки в доказательствах, в трактовке понятий и категорий, проявляет низкую культуру знаний, не владеет основными умениями и навыками при выполнении практических задач. Студент отказывается от ответов на дополнительные вопросы

Лист изменений и дополнений

№ п/п	Виды дополнений и изменений	Дата и номер протокола заседания кафедры (кафедр), на котором были рассмотрены и одобрены изменения и дополнения	Подпись (с расшифровкой) заведующего кафедрой (заведующих кафедрами)

Экспертное заключение

Представленный фонд оценочных средств (далее – ФОС) по дисциплине «Объектно-ориентированное программирование» соответствует требованиям ФГОС ВО.

Предлагаемые формы и средства текущего и промежуточного контроля адекватны целям и задачам реализации основной профессиональной образовательной программы по направлению подготовки 44.03.04. Профессиональное обучение (по отраслям).

Оценочные средства для текущего контроля успеваемости, промежуточной аттестации по итогам освоения дисциплины представлены в полном объеме.

Виды оценочных средств, включенные в представленный фонд, отвечают основным принципам формирования ФОС.

Разработанный и представленный для экспертизы фонд оценочных средств рекомендуется к использованию в процессе подготовки обучающихся по указанному направлению.

Председатель учебно-методической комиссии факультета компьютерных систем и информационных технологий — Ветрова Н. Н.