Областное государственное образовательное учреждение среднего профессионального образования «Иркутский авиационный техникум»

«УТВЕРЖДАЮ»
Директор
ОГБОУ СПО "ИАТ"
_____ В.Г. Семенов
«31» августа 2013 г.

Фонд оценочных средств
по профессиональному модулю
ПМ.01 Разработка программных модулей программного обеспечения для компьютерных систем
образовательной программы
по специальности СПО
09.02.03 Программирование в компьютерных системах

г.Иркутск

Рассмотрена
цикловой комиссией
_____
Протокол № _____
от «___» ____ 20__ г.

Председатель ЦК
_____ /_____ /

№	Разработчик ФИО (полностью)
1	Некипелова Альбина Сергеевна

1. ОБЩИЕ ПОЛОЖЕНИЯ

1.1. Область применения фонда оценочных средств (ФОС)
ФОС профессионального модуля – является частью образовательной программы в соответствии с ФГОС СПО по специальности 09.02.03 Программирование в компьютерных системах
в части освоения вида профессиональной деятельности (ВПД):
Разработка программных модулей программного обеспечения для компьютерных систем
и соответствующих профессиональных компетенций (ПК):

ПК.1.1 Выполнять разработку спецификаций отдельных компонент.
ПК.1.2 Осуществлять разработку кода программного продукта на основе готовых спецификаций на уровне модуля.
ПК.1.3 Выполнять отладку программных модулей с использованием специализированных программных средств.
ПК.1.4 Выполнять тестирование программных модулей.
ПК.1.5 Осуществлять оптимизацию программного кода модуля.
ПК.1.6 Разрабатывать компоненты проектной и технической документации с использованием графических языков спецификаций.

1.2 Цели и задачи модуля – требования к результатам освоения модуля
С целью овладения указанным видом профессиональной деятельности и соответствующими профессиональными компетенциями обучающийся в ходе освоения профессионального модуля должен:

В результате освоения дисциплины обучающийся должен	№ дидактической единицы	Формируемая дидактическая единица
Знать	1.1	основные этапы разработки программного обеспечения;
	1.2	основные принципы технологии структурного и объектно-ориентированного программирования;
	1.3	основные принципы отладки и тестирования программных продуктов;
	1.4	методы и средства разработки технической документации

Уметь	2.1	осуществлять разработку кода программного модуля на современных языках программирования;
	2.2	создавать программу по разработанному алгоритму как отдельный модуль;
	2.3	выполнять отладку и тестирование программы на уровне модуля;
	2.4	оформлять документацию на программные средства;
	2.5	использовать инструментальные средства для автоматизации оформления документации;
Иметь практический опыт	3.1	разработки алгоритма поставленной задачи и реализации его средствами автоматизированного проектирования;
	3.2	разработки кода программного продукта на основе готовой спецификации на уровне модуля;
	3.3	использования инструментальных средств на этапе отладки программного продукта;
	3.4	проведения тестирования программного модуля по определенному сценарию;

2. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ МЕЖДИСЦИПЛИНАРНЫХ КУРСОВ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ ИСПОЛЬЗУЕМЫЙ НА ТЕКУЩЕМ КОНТРОЛЕ

2.1 Результаты освоения МДК.01.01 подлежащие проверке на текущем контроле

№ текущего контроля	Индекс занятия текущего контроля	Проверяемая дидактическая единица	Формируемые компетенции	Основные показатели оценивания результата	№ задания относящийся к показателю оценивания	Метод контроля	Форма контроля	Вид контроля	Индексы занятий ранее изученных связанные с контролируемыми дидактическими единицами
1	1.1.3	1.1	ПК.1.1	1.1.1.1, 1.1.1.2	1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8	Опрос	Практическая работа	Защита	1.1.1, 1.1.2
		2.1	ПК.1.1	1.2.1.1	1.1, 1.7, 1.8	Опрос	Практическая работа	Защита	1.1.1, 1.1.2
2	1.1.5	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.1.1.1	2.1, 2.2	Опрос	Практическая работа	Защита	1.1.3, 1.1.4
		2.1	ПК.1.1, ПК.1.2,	2.2.1.1	2.1, 2.2	Опрос	Практическая работа	Защита	1.1.3, 1.1.4

			ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6						
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.2.2.1	2.1, 2.2	Опрос	Практическая работа	Защита	1.1.4
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.2.3.1	2.1, 2.2	Опрос	Практическая работа	Защита	1.1.4
3	1.1.7	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.1.1.1	3.1	Опрос	Практическая работа	Защита	1.1.5, 1.1.6
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4,	3.2.1.1	3.1	Опрос	Практическая работа	Защита	1.1.5, 1.1.6

			ПК.1.5, ПК.1.6						
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.2.2.1	3.1	Опрос	Практическая работа	Защита	1.1.6
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.2.3.1	3.1	Опрос	Практическая работа	Защита	1.1.6
4	1.1.9	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	4.1.1.1	4.1	Опрос	Практическая работа	Защита	1.1.7, 1.1.8
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	4.2.1.1	4.1	Опрос	Практическая работа	Защита	1.1.7, 1.1.8

		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	4.2.2.1	4.1	Опрос	Практическая работа	Защита	1.1.7, 1.1.8
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	4.2.3.1	4.1	Опрос	Практическая работа	Защита	1.1.7, 1.1.8
5	1.1.11	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	5.1.1.1	5.1	Опрос	Практическая работа	Защита	1.1.9, 1.1.10
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	5.2.1.1	5.1	Опрос	Практическая работа	Защита	1.1.9, 1.1.10
6	1.1.13	1.1	ПК.1.1, ПК.1.2,	6.1.1.1	6.1	Опрос	Практическая работа	Защита	1.1.11, 1.1.12

			ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6						
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	6.2.1.1	6.1	Опрос	Практическая работа	Защита	1.1.11, 1.1.12
7	1.2.2	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	7.1.1.1	7.1	Опрос	Практическая работа	Защита	1.1.13, 1.1.14, 1.1.15, 1.2.1
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	7.2.1.1	7.1	Опрос	Практическая работа	Защита	1.1.13, 1.1.14, 1.1.15, 1.2.1
8	1.2.4	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4,	8.1.1.1	8.1	Опрос	Практическая работа	Защита	1.2.2, 1.2.3

			ПК.1.5, ПК.1.6						
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	8.2.1.1	8.2	Опрос	Практическая работа	Защита	1.2.2, 1.2.3
9	1.2.5	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	9.1.1.1	9.1	Опрос	Практическая работа	Защита	1.2.4
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	9.1.2.1	9.2	Опрос	Практическая работа	Защита	1.1.6, 1.1.7, 1.1.8, 1.1.9, 1.1.10, 1.1.11, 1.1.12, 1.1.13, 1.1.14, 1.1.15, 1.2.1, 1.2.2, 1.2.3, 1.2.4
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	9.1.3.1	9.2, 9.3	Опрос	Практическая работа	Защита	1.1.4, 1.1.6, 1.1.7, 1.1.8, 1.1.9, 1.1.10, 1.1.11, 1.1.12, 1.1.13, 1.1.14,

			ПК.1.6					1.1.15, 1.2.1, 1.2.2, 1.2.3, 1.2.4
		1.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	9.1.4.1	9.2	Опрос	Практическая работа	Защита	1.2.4
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	9.2.1.1	9.3	Опрос	Практическая работа	Защита	1.2.4
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	9.2.2.1	9.1, 9.2, 9.3	Опрос	Практическая работа	Защита	1.1.9, 1.1.10, 1.1.11, 1.1.12, 1.1.13, 1.1.14, 1.1.15, 1.2.1, 1.2.2, 1.2.3, 1.2.4
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	9.2.3.1	9.1, 9.2, 9.3	Опрос	Практическая работа	Защита	1.1.9, 1.1.10, 1.1.11, 1.1.12, 1.1.13, 1.1.14, 1.1.15, 1.2.1, 1.2.2, 1.2.3,

			ПК.1.6						1.2.4
		2.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	9.2.4.1	9.1, 9.2, 9.3	Опрос	Практическая работа	Защита	1.2.1, 1.2.4
		2.5	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	9.2.5.1	9.1, 9.2, 9.3	Опрос	Практическая работа	Защита	1.2.1, 1.2.4
10	1.2.6	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	10.1.1.1	10.1, 10.2, 10.3	Опрос	Практическая работа	Защита	1.2.5
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	10.2.1.1	10.1, 10.2, 10.3	Опрос	Практическая работа	Защита	1.2.5
11	1.2.8	1.1	ПК.1.1,	11.1.1.1	11.1, 11.2,	Опрос	Практическая	Защита	1.2.6, 1.2.7

			ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6		11.3, 11.4, 11.5, 11.6		работа		
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	11.2.1.1	11.1, 11.2, 11.3, 11.4, 11.5, 11.6	Опрос	Практическая работа	Защита	1.2.6, 1.2.7
12	1.2.10	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	12.1.1.1	12.1	Сравнение с аналогом	Практическая работа	Защита	1.2.8, 1.2.9
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	12.2.1.1	12.1	Сравнение с аналогом	Практическая работа	Защита	1.2.8, 1.2.9
13	1.2.14	1.1	ПК.1.1, ПК.1.2, ПК.1.3,	13.1.1.1, 13.1.1.2	13.1, 13.2, 13.3	Опрос	Практическая работа	Защита	1.2.10, 1.2.11, 1.2.12, 1.2.13

			ПК.1.4, ПК.1.5, ПК.1.6						
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	13.2.1.1, 13.2.1.2	13.1, 13.2, 13.3	Опрос	Практическая работа	Защита	1.2.10, 1.2.11, 1.2.12, 1.2.13
14	1.2.17	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	14.1.1.1, 14.1.1.2	14.1, 14.2, 14.3, 14.4	Сравнение с аналогом	Практическая работа	Защита	1.2.14, 1.2.15, 1.2.16
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	14.2.1.1, 14.2.1.2	14.1, 14.2, 14.3, 14.4	Сравнение с аналогом	Практическая работа	Защита	1.2.14, 1.2.15, 1.2.16

Перечень заданий текущего контроля

Номер задания	Задания

1.1	Перевести числа из десятичной системы в двоичную **12, 45, 123, 372, 512, 1024**
1.2	Перевести числа из десятичной системы в шестнадцатиричную **15, 23, 85, 345, 1084**
1.3	Перевести числа из двоичной системы в десятичную **0001 B, 0001 0111 B, 0010 0101 B**
1.4	Перевести числа из двоичной системы в шестнадцатиричную. **0011 B, 0111 0010 B, 0111 0000 B**
1.5	Перевести числа из шестнадцатиричной системы в двоичную **FB, 01 C4, A5 E7**
1.6	Перевести числа из шестнадцатиричной системы в десятичную **C6, A3 FD, 01 C2**
1.7	Как будут выглядеть в памяти машины IBM PC числа и символы -328,1110011101101001b, 95 , @, { если они расположатся там, начиная с адреса FFEC. Представить числа в виде 2-х байтов
1.8	Составить программу в машинных кодах:

	-занести в регистр AX десятичное число -184 . -прибавить десятичное число 15 к AX. -переслать содержимое AX в BX. -прибавить AX к BX. -почистить AX. -выход в DOS. Записать программу в машинных кодах в память со смещением 100 Рассмотреть содержимое всех регистров. Рассмотреть записанную программу в памяти. Осуществить пошаговое выполнение созданной программы до команды RET
2.1	Составить программу в машинных кодах: - переслать слово(число 34)- два байта, начинающиеся в сегменте данных с адреса 04 в регистр AX, -прибавить содержимое слова(второе число 12)- два байта,начинающиеся в

	сегменте данных с адреса 02 к регистру AX, -переслать содержимое регистра AX в слово ,начинающиеся в сегменте данных DS с адреса 00, -вернуться в DOS. Рассмотреть содержимое сегмента данных . Рассмотреть записанную программу в памяти. Рассмотреть содержимое всех регистров. Осуществить пошаговое выполнение созданной программы до команды RET.
2.2	1Написать программу на языке программирования Ассемблер, которая заносит число 5 в регистры AX, BX, CX, DX. Создать объектный, выполняемый файл просмотреть EXE файл в отладчике.
3.1	1.Составить программу на языке АССЕМБЛЕРА ,задавая все определения с учетом того, что выполняемый модуль должен иметь расширение .EXE . -в сегменте данных задать следующие числа и символьные выражения: Фамилия, через запятую-Имя,Отчество,возраст,

номер дома,

номер квартиры,

любое двоичное число>16,

любое шестнадцатеричное число в интервале от 10 до 1000,

по возможности задать четыре последних числа в одном байте,

в двух байтах,

в четырех байтах,

в восьми байтах,

в десяти байтах.

-задать любым трем числам произвольные метки ,а в словах с именами ADR1 , ADR2 , ADR3 определить адреса этих чисел. В сегменте данных любое число кроме последнего обозначить меткой MMM.

2.Тело программы должно содержать следующее:

-в регистр CX непосредственно занести номер дома;

-в регистр BX занести второе слово содержащееся за меткой MMM;

-в регистр AX занести число или данное находящееся по адресу ADR2

используя косвенную адресацию;

-считая что все эти данные числовые получить их сумму в регистрах

DX и AL;

-полученную сумму занести в сегмент данных в байт LL и в слово

XX ;

-выход в DOS.

3.Последовательно получить : файл с расширением .asm ,

файл с расширением .obj ,

файл с расширением .lst ,

файл с расширением .exe .

используя любой редактор и программы MASM.EXE, LINK.EXE

4.1	Написать программу на языке программирования Ассемблер: сложит два числа, находящиеся по адресу **pp** и **pp1,** результат занести по адресу **sum.** Последовательно получить : файл с расширением .asm , файл с расширением .obj , файл с расширением .lst , файл с расширением .exe . Прорешать созданную программу в DEBUG, найти сегмент данных, сегмент кодов, сегмент стека, связывая данные в листинге с данными в памяти. Отчет по листингу или в отладчике. Изменить созданную программу написанную на АССЕМБЛЕР'е таким образом, чтобы получить исходный модуль с расширением .COM
5.1	Составить программу на языке АССЕМБЛЕРА, задавая все определения с учетом того, что выполняемый модуль должен иметь расширение .EXE. В сегменте данных любое число кроме последнего обозначить меткой MMM. 2. Тело программы должно содержать следующее: - в регистр CX непосредственно занести номер дома;

- в регистр BX занести второе слово содержащееся за меткой MMM;

- в регистр AX занести число или данное находящееся по адресу ADR2 используя косвенную адресацию;

- считая, что все эти данные числовые получить их сумму в регистре DX;

- полученную сумму занести в сегмент данных;

- конец.

3. Последовательно получить : файл с расширением .asm ,

файл с расширением .obj ,

файл с расширением .lst ,

файл с расширением .exe .

4. Прорешать созданную программу в DEBUG, найти сегмент данных, сегмент кодов, сегмент стека, связывая данные в листинге с данными в памяти.

5. Работать только в своей директории .

	6. Отчет по листингу или в отладчике. 7. Изменить созданную программу написанную на АССЕМБЛЕР'е таким образом, чтобы получить исходный модуль с расширением .COM 8. Поставить файл с расширением .COM на выполнение
6.1	1Написать программу на языке программирования Ассемблер, которая заносит число 5 в регистры AX, BX, CX, DX. Создать объектный, выполняемый файл просмотреть EXE файл в отладчике. **Задание** 2 1.Составить программу на языке АССЕМБЛЕРА ,задавая все определения с учетом того, что выполняемый модуль должен иметь расширение .EXE . -в сегменте данных задать следующие числа и символьные выражения: Фамилия, через запятую-Имя,Отчество,возраст, номер дома, номер квартиры, любое двоичное число>16,

любое шестнадцатеричное число в интервале от 10 до 1000,

по возможности задать четыре последних числа в одном байте,

в двух байтах,

в четырех байтах,

в восьми байтах,

в десяти байтах.

-задать любым трем числам произвольные метки ,а в словах с именами ADR1 , ADR2 , ADR3 определить адреса этих чисел. В сегменте данных любое число кроме последнего обозначить меткой MMM.

2.Тело программы должно содержать следующее:

-в регистр CX непосредственно занести номер дома;

-в регистр BX занести второе слово содержащееся за меткой MMM;

-в регистр AX занести число или данное находящееся по адресу ADR2 используя косвенную адресацию;

	-считая что все эти данные числовые получить их сумму в регистрах DX и AL; -полученную сумму занести в сегмент данных в байт LL и в слово XX ; -выход в DOS. 3.Последовательно получить : файл с расширением .asm , файл с расширением .obj , файл с расширением .lst , файл с расширением .exe . используя любой редактор и программы MASM.EXE, LINK.EXE
7.1	1Написать программу на языке программирования Ассемблер, которая заносит число 5 в регистры AX, BX, CX, DX. Создать объектный, выполняемый файл просмотреть EXE файл в отладчике. **Задание** 2 1.Составить программу на языке АССЕМБЛЕРА ,задавая все определения с учетом того, что

выполняемый модуль должен иметь расширение .EXE .

-в сегменте данных задать следующие числа и символьные выражения:

Фамилия, через запятую-Имя,Отчество,возраст,

номер дома,

номер квартиры,

любое двоичное число>16,

любое шестнадцатеричное число в интервале от 10 до 1000,

по возможности задать четыре последних числа в одном байте,

в двух байтах,

в четырех байтах,

в восьми байтах,

в десяти байтах.

-задать любым трем числам произвольные метки ,а в словах с именами

ADR1 , ADR2 , ADR3 определить адреса этих чисел. В сегменте дан-

ных любое число кроме последнего обозначить меткой MMM.

2.Тело программы должно содержать следующее:

-в регистр CX непосредственно занести номер дома;

-в регистр BX занести второе слово содержащееся за меткой MMM;

-в регистр AX занести число или данное находящееся по адресу ADR2

используя косвенную адресацию;

-считая что все эти данные числовые получить их сумму в регистрах

DX и AL;

-полученную сумму занести в сегмент данных в байт LL и в слово

XX ;

-выход в DOS.

3.Последовательно получить : файл с расширением .asm ,

файл с расширением .obj ,

файл с расширением .lst ,

	файл с расширением .exe . используя любой редактор и программы MASM.EXE, LINK.EXE
8.1	1Написать программу на языке программирования Ассемблер. Дан ряд чисел. Если в характеристике XAR появляется определённое число, то числа из буфера складываются, в противоположном случае они вычитаются
8.2	Составить программу обнуления памяти (любой области, заданной в сегменте данных) в размере 10 шестнадцатиразрядных слов. Задать буфер следующим образом: **buf db 20 dup('*')**
9.1	1Написать программу на языке программирования Ассемблер. Дан ряд чисел. Если в характеристике XAR появляется определённое число, то числа из буфера складываются, в противоположном случае они вычитаются
9.2	Составить программу обнуления памяти (любой области, заданной в сегменте данных) в размере 10 шестнадцатиразрядных слов. Задать

	буфер следующим образом: **buf db 20 dup('*')**
9.3	1Составить программу занесения в каждый байт буфера размером 25 байт числа - 0FCH.
10.1	1Составить программу занесения в каждый байт буфера размером 25 байт числа - 0FCH.
10.2	Составить программу занесения в память последовательной цепочки чисел (1 ,2 ,3 и т.д. до 16), учитывая ,что каждое число занимает 2-а байта памяти.
10.3	Составить программу занесения в память последовательной цепочки чисел (100 ,99 ,98 и т.д. до 1), учитывая ,что каждое число занимает 2-а байта памяти.
11.1	Составить программу заполнения буфера размером 34 байта числом 13 ,начиная формировать буфер с конца,как формируется стек
11.2	Составить программу заполнения буфера следущими цепочками символов (@ # &),буфер взять равным 36 байтам .
11.3	Составить программу занесения в память последовательной цепочки чисел (50 ,0 ,40 ,0, 30 ,0 и т.д. до 0 ,0), учитывая ,что каждое число

занимает 1 байта памяти.

11.4	Составить программу занесения в память последовательной цепочки символов от a до z
11.5	Составить программу на АССЕМБЛЕР'е ,как программу с расширением .exe , введя в качестве данных одно следующее чисело 11000011 10001111 2. Заполнить буфер памяти в байтах= (14 * 4) символами #, если бит предложенного слова равный14в двоичном слове будет=1 и заполнить этот же буфер нулями ,если рассмотренный бит =0. 3. Выделить этот бит и в качестве 1 или 0 записать в регистр dx. 4. Поместить за заполненным буфером два символа @@ , если анализируемое число положительно ,иначе поместить два символа &&. 5. Прорешать созданную программу в DEBUG'е ,меняя анализируемое число
11.6	1.Изменить предыдущую программу таким образом : 2. Если бит предложенного слова равный 14в двоичном слове =1,то выдать сообщение "Бит равен единице" ,если рассмотренный бит =0 то

	выдать сообщение "Бит равен нулю". 3. Выделить этот бит и в качестве 1 или 0 записать в регистр dx. 4. Проанализировать знак числа, если предложенное число >0 ,выдать сообщение "Число положительное", иначе выдать сообщение "Число отрицательное" . 5.Прорешать созданную программу в DOS
12.1	Создать com-файл. Расписать в тетради, как происходит сдвиг битов в байтах при разных сдвигах. **CCC SEGMENT** **assume DS:CCC,CS:CCC,SS:CCC** **ORG 100H** **VX: JMP PP** **FIF DB 5** **PP PROC NEAR** **MOV AL,FIF**

```
SHR     AL,1
MOV     AL,-5
SHR     AL,1

MOV     AL,FIF
SHL     AL,1
MOV     AL,-5
SHL     AL,1

MOV     AL,FIF
SAR     AL,1
MOV     AL,-5
SAR     AL,1
```

	MOV AL,FIF **SAL AL,1** **MOV AL,-5** **SAL AL,1** **RET** **pp ENDP** **ccc ENDS** **END VX**
13.1	**Задание** Создать COM-файл **title Primer1** **add segment para** **assume ss:add,ds:add,cs:add**

```
        org 100h
vv:     jmp nach
nam1    db '123456'
nam2    db 'abcdef'

nach    proc      near
        lea       si,nam1
        lea       di,nam2
        mov       cx,6
m1:
        mov       al,[si]
        mov       ah,[di]
        mov       [si],ah
        mov       [di],al
```

	inc si **inc di** **dec cx** **jnz m1** **ret** **nach endp** **add ends** **end vv**
13.2	Ввести строку символьных данных ,задавая буфер равный 30 байт. Подсчитать в этой строке количество символов "f" . Выдать одно из сообщений: "Символа f в строке данных нет" "Символ f встречается 1 раз" "В строке данных символов f >=2".
13.3	Ввести строку символьных данных ,задавая буфер равный 40 байт.

	Заменить в этой строке встречающийся символ "d" на символ "s", Выдать полученную строку символов в первую строку экрана,на- чиная с 10 позиции.
14.1	Создать exe-файл **Q10 SEGMENT PARA** **ASSUME SS:Q10,DS:Q10,CS:Q10** **ORG 100H** **Q11: JMP Q22** **NN1 DW 4860H** **KOD DB ?** **COO1 DB ' Не готово устройство','$'** **COO2 DB ' Готово для ввода','$'** **COO3 DB ' Готово для вывода','$'**

```
COO4  DB ' Сбой','$'
Q22   PROC NEAR
      MOV  DX,NN1
      IN   AL,DX
      MOV  KOD,AL
      AND  AL,11001111B
      CMP  AL,00H
      JNE  B10
      LEA  DX,COO1
      JMP  B30
B10:  CMP  AL,10000B
      JNE  B20
      LEA  DX,COO2
      JMP  B30
```

```
B20:  CMP  AL,100000B
      JNE  B20
      LEA  DX,COO3
      JMP  B30
      LEA  DX,COO4
B30:  MOV  AH,09
      INT  21H
      MOV  DX,NN1
      MOV  AL,KOD
      AND  AL,11001111B
      OUT  DX,AL
      RET
Q22   ENDP
Q10   ENDS
```

	END Q11
14.2	Ввести строку символьных данных ,задавая буфер равный 10 байт. Проанализировать встречающиеся символы . Выдать одно из сообщений: "Символы русского регистра" "Символы латинского регистра" "Символы и русского и латинского регистров".
14.3	Ввести одну из строк символьных данных : "единица","два ", "три ","четыре ","пять ","шесть ","семь ","восемь ", "девять " - проанализировав введенные данные выдать на экран 1 или 2 или 3 или 4 или 5 или 6 или 7 или 8 или 9.
14.4	Ввести строку символьных данных ,задавая буфер равный 12 байт. Переставить символы в строке следующим образом :первый символ на место последнего , второй символ на место предпоследнего ,

	..,предпоследний на место второго,а последний на место первого. Выдать полученную строку символов в 10 строку экрана,на- чиная с 30 позиции.

Перечень показателей текущего контроля

Номер показателя	Значение показателя
1.1.1.1	Знание перевода чмсел из одной системы счисления в другую
1.1.1.2	Команды отладчика Debug.Представление в памяти различных типов данных
1.2.1.1	Команды отладчика Debug.Представление в памяти различных типов данных
2.1.1.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и выполнению программ
2.2.1.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и выполнению программ
2.2.2.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и выполнению программ
2.2.3.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и выполнению программ
3.1.1.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и

	выполнению программ Работа с директивами памяти
3.2.1.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и выполнению программ Работа с директивами памяти
3.2.2.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и выполнению программ Работа с директивами памяти
3.2.3.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и выполнению программ Работа с директивами памяти
4.1.1.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и выполнению программ
4.2.1.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и выполнению программ
4.2.2.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и выполнению программ
4.2.3.1	Закрепление на практике теоретические знания по вводу, ассемблированию, компоновке и выполнению программ
5.1.1.1	Знание директив Ассемблера
5.2.1.1	Знание директив Ассемблера

6.1.1.1	Знание основных команд процессора
6.2.1.1	Знание основных команд процессора
7.1.1.1	Знание структуры EXE - программы
7.2.1.1	Знание структуры EXE - программы
8.1.1.1	Знание команд организации разветвлений
8.2.1.1	Знание команд организации разветвлений
9.1.1.1	Знание разветвлений и циклов
9.1.2.1	Знание разветвлений и циклов
9.1.3.1	Знание разветвлений и циклов
9.1.4.1	Знание разветвлений и циклов
9.2.1.1	Знание разветвлений и циклов
9.2.2.1	Знание разветвлений и циклов
9.2.3.1	Знание разветвлений и циклов
9.2.4.1	Знание разветвлений и циклов
9.2.5.1	Знание разветвлений и циклов
10.1.1.1	Знание команд организации циклов
10.2.1.1	Знание команд организации циклов
11.1.1.1	Знание команд логических операций
11.2.1.1	Знание команд логических операций
12.1.1.1	Знание команд сдвигов
12.2.1.1	Знание команд сдвигов
13.1.1.1	Знание структуры COM - программ

13.1.1.2	Знание работы с экраном и курсором
13.2.1.1	Умение создавать COM -программу
13.2.1.2	Умение работать с экраном и курсором
14.1.1.1	Знание работы с портами ввода - вывода
14.1.1.2	Знание содания программ с использованием ввода - вывода на экран
14.2.1.1	Умение работы с портами ввода - вывода
14.2.1.2	Умение содавать программы с использованием ввода - вывода на экран

2.2 Результаты освоения МДК.01.02 подлежащие проверке на текущем контроле

№ текущего контроля	Индекс занятия текущего контроля	Проверяемая дидактическая единица	Формируемые компетенции	Основные показатели оценивания результата	№ задания относящийся к показателю оценивания	Метод контроля	Форма контроля	Вид контроля	Индексы занятий ранее изученных связанные с контролируемыми дидактическими единицами
1	1.1.6	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4	1.1.1.1, 1.1.1.2	1.1, 1.2	Опрос	Самостоятельная работа	Защита	1.1.1, 1.1.2, 1.1.3, 1.1.4, 1.1.5
		2.1	ПК.1.1, ПК.1.2, ПК.1.3,	1.2.1.1	1.1, 1.2	Опрос	Практическая работа	Защита	1.1.1, 1.1.2, 1.1.3, 1.1.4, 1.1.5

			ПК.1.4						
2	1.1.9	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5	2.1.1.1	2.1, 2.2	Опрос	Практическая работа	Защита	1.1.6, 1.1.7, 1.1.8
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5	2.1.2.1	2.1, 2.2	Опрос	Практическая работа	Защита	1.1.3, 1.1.4, 1.1.5, 1.1.6, 1.1.7, 1.1.8
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5	2.2.1.1	2.1, 2.2	Опрос	Практическая работа	Защита	1.1.6, 1.1.7, 1.1.8
3	1.1.12	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5	3.1.1.1	3.1, 3.2	Опрос	Практическая работа	Защита	1.1.9, 1.1.10, 1.1.11
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5	3.1.2.1	3.1, 3.2	Опрос	Практическая работа	Защита	1.1.9, 1.1.10, 1.1.11

		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5	3.2.1.1	3.1, 3.2	Опрос	Практическая работа	Защита	1.1.9, 1.1.10, 1.1.11
4	1.1.14	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5	4.1.1.1	4.1, 4.2, 4.3	Опрос	Практическая работа	Защита	1.1.12, 1.1.13
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5	4.1.2.1	4.1, 4.2, 4.3	Опрос	Практическая работа	Защита	1.1.12, 1.1.13
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5	4.2.1.1	4.1, 4.2, 4.3	Опрос	Практическая работа	Защита	1.1.12, 1.1.13
5	1.1.16	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5	5.1.1.1	5.1	Опрос	Практическая работа	Защита	1.1.14, 1.1.15
		1.2	ПК.1.1,	5.1.2.1	5.1	Опрос	Практическая	Защита	1.1.14, 1.1.15

			ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5				работа		
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5	5.2.1.1	5.1	Опрос	Практическая работа	Защита	1.1.14, 1.1.15
6	1.1.18	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	6.1.1.1	6.1, 6.2	Опрос	Практическая работа	Защита	1.1.16, 1.1.17
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	6.1.2.1	6.3, 6.4	Опрос	Практическая работа	Защита	1.1.16, 1.1.17
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	6.1.3.1	6.1, 6.2, 6.3, 6.4	Опрос	Практическая работа	Защита	1.1.6, 1.1.8, 1.1.9, 1.1.10, 1.1.12, 1.1.13, 1.1.14, 1.1.15, 1.1.16, 1.1.17

			ПК.1.6					
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	6.2.1.1	6.1, 6.2, 6.3, 6.4	Опрос	Практическая работа	Защита	1.1.16, 1.1.17
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	6.2.2.1	6.1, 6.2, 6.3, 6.4	Опрос	Практическая работа	Защита	1.1.3, 1.1.4, 1.1.5, 1.1.6, 1.1.7, 1.1.8, 1.1.9, 1.1.10, 1.1.11, 1.1.12, 1.1.13, 1.1.14, 1.1.15, 1.1.16, 1.1.17
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	6.2.3.1	6.1, 6.2, 6.3, 6.4	Опрос	Практическая работа	Защита	1.1.3, 1.1.6, 1.1.8, 1.1.9, 1.1.10, 1.1.11, 1.1.12, 1.1.13, 1.1.14, 1.1.15, 1.1.16, 1.1.17
		2.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	6.2.4.1	6.1, 6.2, 6.3, 6.4	Опрос	Практическая работа	Защита	1.1.16

			ПК.1.6						
		2.5	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	6.2.5.1	6.1, 6.2, 6.3, 6.4	Опрос	Практическая работа	Защита	1.1.16
7	1.1.20	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	7.1.1.1	7.1	Опрос	Практическая работа	Защита	1.1.18, 1.1.19
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	7.1.2.1	7.1	Опрос	Практическая работа	Защита	1.1.18, 1.1.19
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	7.1.3.1	7.1	Опрос	Практическая работа	Защита	1.1.18, 1.1.19
		2.1	ПК.1.1,	7.2.1.1	7.1	Опрос	Практическая	Защита	1.1.18, 1.1.19

			ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6				работа		
8	1.1.22	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	8.1.1.1	8.1	Опрос	Практическая работа	Защита	1.1.20, 1.1.21
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	8.1.2.1	8.1	Опрос	Практическая работа	Защита	1.1.20, 1.1.21
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	8.1.3.1	8.1	Опрос	Практическая работа	Защита	1.1.20, 1.1.21
		2.1	ПК.1.1, ПК.1.2, ПК.1.3,	8.2.1.1	8.1	Опрос	Практическая работа	Защита	1.1.20, 1.1.21

			ПК.1.4, ПК.1.5, ПК.1.6						
9	1.1.24	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	9.1.1.1	9.1	Опрос	Практическая работа	Защита	1.1.22, 1.1.23
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	9.1.2.1	9.1	Опрос	Практическая работа	Защита	1.1.22, 1.1.23
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	9.1.3.1	9.1	Опрос	Практическая работа	Защита	1.1.22, 1.1.23
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	9.2.1.1	9.1	Опрос	Практическая работа	Защита	1.1.22, 1.1.23

			ПК.1.6						
10	1.1.26	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	10.1.1.1	10.1	Опрос	Практическая работа	Защита	1.1.24, 1.1.25
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	10.1.2.1	10.1, 10.2	Опрос	Практическая работа	Защита	1.1.24, 1.1.25
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	10.1.3.1	10.1, 10.2	Опрос	Практическая работа	Защита	1.1.24, 1.1.25
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	10.2.1.1	10.1, 10.2	Опрос	Практическая работа	Защита	1.1.24, 1.1.25
11	1.2.3	1.1	ПК.1.1,	11.1.1.1	11.1	Сравнение с	Практическая	Защита	1.1.26, 1.2.1,

			ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6			аналогом	работа		1.2.2
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	11.1.2.1	11.1	Сравнение с аналогом	Практическая работа	Защита	1.1.26, 1.2.1, 1.2.2
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	11.1.3.1	11.1	Сравнение с аналогом	Практическая работа	Защита	1.1.26, 1.2.1, 1.2.2
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	11.2.1.1	11.1	Сравнение с аналогом	Практическая работа	Защита	1.1.26, 1.2.1, 1.2.2
		2.2	ПК.1.1, ПК.1.2, ПК.1.3,	11.2.2.1	11.1	Сравнение с аналогом	Практическая работа	Защита	1.1.18, 1.1.19, 1.1.20, 1.1.21, 1.1.22, 1.1.23,

			ПК.1.4, ПК.1.5, ПК.1.6						1.1.24, 1.1.25, 1.1.26, 1.2.1, 1.2.2
12	1.2.5	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	12.1.1.1	12.1	Сравнение с аналогом	Практическая работа	Защита	1.2.3, 1.2.4
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	12.1.2.1	12.1	Сравнение с аналогом	Практическая работа	Защита	1.2.3, 1.2.4
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	12.1.3.1	12.1	Сравнение с аналогом	Практическая работа	Защита	1.2.3
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	12.2.1.1	12.1	Сравнение с аналогом	Практическая работа	Защита	1.2.3, 1.2.4

			ПК.1.6						
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	12.2.2.1	12.1	Сравнение с аналогом	Практическая работа	Защита	1.2.3, 1.2.4
13	1.2.7	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	13.1.1.1	13.1	Сравнение с аналогом	Практическая работа	Защита	1.2.5, 1.2.6
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	13.1.2.1	13.1	Сравнение с аналогом	Практическая работа	Защита	1.2.5, 1.2.6
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	13.1.3.1	13.1	Сравнение с аналогом	Практическая работа	Защита	
		2.1	ПК.1.1,	13.2.1.1	13.1	Сравнение с	Практическая	Защита	1.2.5, 1.2.6

			ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6			аналогом	работа		
14	1.2.9	1.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	14.1.1.1	14.1, 14.2, 14.3	Сравнение с аналогом	Практическая работа	Защита	1.2.7, 1.2.8
		1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	14.1.2.1	14.1, 14.2, 14.3	Сравнение с аналогом	Практическая работа	Защита	1.2.7, 1.2.8
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	14.1.3.1	14.1, 14.2, 14.3	Сравнение с аналогом	Практическая работа	Защита	1.2.7
		2.1	ПК.1.1, ПК.1.2, ПК.1.3,	14.2.1.1	14.1, 14.2, 14.3	Сравнение с аналогом	Практическая работа	Защита	1.2.7, 1.2.8

			ПК.1.4, ПК.1.5, ПК.1.6						
15	1.2.11	1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	15.1.2.1	15.1, 15.2, 15.3	Сравнение с аналогом	Практическая работа	Защита	1.2.9, 1.2.10
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	15.2.1.1	15.1, 15.2, 15.3	Сравнение с аналогом	Практическая работа	Защита	1.2.9, 1.2.10
16	1.2.13	1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	16.1.2.1	16.1, 16.2, 16.3	Сравнение с аналогом	Практическая работа	Защита	1.2.11, 1.2.12
		2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	16.2.1.1	16.1, 16.2, 16.3	Сравнение с аналогом	Практическая работа	Защита	1.2.11, 1.2.12

			ПК.1.6						
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	16.2.2.1	16.1, 16.2, 16.3	Сравнение с аналогом	Практическая работа	Защита	1.2.5, 1.2.6, 1.2.7, 1.2.8, 1.2.9, 1.2.10, 1.2.11, 1.2.12
17	1.2.15	1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	17.1.2.1	17.1	Опрос	Индивидуальные задания	Защита	1.2.13, 1.2.14
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	17.1.3.1	17.1	Опрос	Индивидуальные задания	Защита	1.2.9, 1.2.11, 1.2.12, 1.2.13, 1.2.14
		1.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	17.1.4.1	17.1	Опрос	Индивидуальные задания	Защита	1.1.16, 1.1.18, 1.1.20, 1.1.22, 1.2.14
		2.1	ПК.1.1,	17.2.1.1	17.1	Опрос	Индивидуальн	Защита	1.2.13, 1.2.14

			ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6				ые задания		
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	17.2.2.1	17.1	Опрос	Индивидуальные задания	Защита	1.2.13, 1.2.14
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	17.2.3.1	17.1	Опрос	Индивидуальные задания	Защита	1.1.18, 1.1.19, 1.1.20, 1.1.21, 1.1.22, 1.1.23, 1.1.24, 1.1.25, 1.1.26, 1.2.1, 1.2.2, 1.2.3, 1.2.5, 1.2.6, 1.2.7, 1.2.9, 1.2.11, 1.2.12, 1.2.13, 1.2.14
		2.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	17.2.4.1	17.1	Опрос	Индивидуальные задания	Защита	1.1.18, 1.1.20, 1.1.21, 1.1.22, 1.1.26, 1.2.14

			ПК.1.6						
		2.5	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	17.2.5.1	17.1	Опрос	Индивидуальные задания		1.1.18, 1.1.20, 1.1.22, 1.1.26, 1.2.14
18	1.2.16	1.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	18.1.2.1	18.1	Опрос	Индивидуальные задания	Защита	1.2.15
		1.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	18.1.3.1	18.1	Опрос	Индивидуальные задания	Защита	1.2.15
		1.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	18.1.4.1	18.1	Опрос	Индивидуальные задания		1.2.15
		2.1	ПК.1.1,	18.2.1.1	18.1	Опрос	Индивидуальн	Защита	1.2.15

			ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6				ые задания		
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	18.2.2.1	18.1	Опрос	Индивидуальные задания	Защита	1.2.15
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	18.2.3.1	18.1	Опрос	Индивидуальные задания		1.2.15
		2.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	18.2.4.1	18.1	Опрос	Индивидуальные задания		1.2.15
		2.5	ПК.1.1, ПК.1.2, ПК.1.3,	18.2.5.1	18.1	Опрос	Индивидуальные задания	Защита	1.2.15

			ПК.1.4, ПК.1.5, ПК.1.6						

Перечень заданий текущего контроля

Номер задания	Задания
1.1	Разобрать текст программы, написанной на языке С++ **#include <iostream.h>** **#include <math.h>** **#include <stdio.h>** **#define PI 3.14** **void main()** **{float r,v,s;** **cout<<"Vvedite r\n";** **cin>>r;**

	v=(3*PI*r*r*r)/4; **s=4*PI*r*r;** **cout<<"s="<<s<<endl;** **cout<<"v="<<v<<endl;** **}**
1.2	Написать программы на языке C++ 1. Написать программу вычисления площади треугольника, если известны длины двух его сторон и величина угла между этими сторонами. S= a*b*sin(u*p/180)/2 2. Написать программу вычисления сопротивления электрической цепи, состоящей из двух параллельно соединённых сопротивлений R=r1*r2/(r1+r2) 3. Написать программу пересчёта расстояния из вёрст в километры(1верста-это 1066,8 м)

4. Написать программу пересчёта веса из фунтов в килограммы (1фунт- это 405,9 грамма)

5. Написать программу вычисления величины дохода по вкладу. Процентная ставка (% годовых) и время хранения (дней) задаются во время работы программы.

Dochod=summ*stavka/365/100*srok

6.Написать программу, которая преобразует введённое с клавиатуры дробное число в денежный формат. Например, число 12,5 должно быть преобразовано к виду 12 руб. 50 коп.

2.1 Разобрать текст программы, написанной на языке C++

```
#include <conio.h>
#include <stdio.h>          /*Подключение библиотеки математических функции*/
#include <iostream.h>
        void main()
{
    int a;
    int b;
    cout << "Введите a= ";
    cin >> a;
    cout << "Введите b= ";
```

	cin >> b;; **if (a == b) cout << "a = b";** **else {if (a > b) cout << "a > b";** **else cout << "a < b";}** **getch();}**
2.2	2 Написать программы на языке С++ **1.** Написать программу, которая сравнивает два введённых с клавиатуры числа. Программа должна указать, какое число больше или , если числа равны, вывести соответствующее сообщение. **2.** Написать программу, которая проверяет является ли год високосным. **3.** Написать программу решения квадратного уравнения. Программа должна проверять правильность исходных данных и в случае, если коэффициент при второй степени неизвестного равен нулю, выводить соответствующее сообщение. **4.** Проверить на чётность введённое с клавиатуры число **5.** Написать программу, которая после введённого с клавиатуры числа (в диапазоне от 1 до 999), обозначающую денежную единицу, дописывает слово «рубль » в правильной форме. Например 12 рублей, 21 рубль и т.д. **6.** Напишите программу , которая запрашивает у пользователя номер дня недели, затем выводит название дня недели или сообщение об ошибке, если введены неверные данные.

	7. Перевести введённую длину в сантиметры и в дюймы с помощью инструкции switch, причём при переводе в сантиметры вводить символ “c”, а при переводе в дюймы символ “i”
3.1	Разобрать текст программы, написанной на языке C++ **#include <iostream.h>** **#include <conio.h>** **#include <stdio.h>** **#include <stdlib.h>** **main()** **{** **int n,i,k;** **int a[10];** **cout<<"\Введите n:";cin>>n;** **k=0;** **for (i=1;i<=n;i++)**

	{ **cout<<"\Введите число";cin>>a[i];** **if (a[i]/3=0)&&(a[i]/5<>0)** **then k=k+1;}** **cout <<"\Количество чисел делющихся на 3 и не делющихся на 5"<<k;** **getch();** **}**
3.2	Написать программы на языке С++ 1.Написать программу, которая выводит таблицу квадратов десяти первых положительных чисел. 2.Написать программу, которая вычисляет сумму первых членов ряда: 1,3,5,7,……Количество суммируемых членов ряда задаётся во время работы программы. 3. Написать программу, которая выводит таблицу степеней двойки от нулевой до десятой.

	4. Написать программу, которая выводит таблицу значений функции **y=-2,4x ^{2} +5x-3** в диапазоне от **–2** до **+2,** с шагом **0,5**
4.1	Разобрать текст программы, написанной на языке C++ **#include <iostream.h>** **#include <conio.h>** **#include <math.h>** **int main()** **{** **float s,e,a;** **int n;** **cout<<"Enter E: ";** **cin>>e;** **s=0;**

```
a=e;
n=1;
while (fabs(a)>=e)
  {
  a=(float)1/((3*n-2)*(3*n+1));
  s+=a;
  n+=1;
  }
cout<<"S="<<s;
getch();
return 0;
}
```

4.2 Написать программы на языке С++

Задачи на циклы do while

	1. Написать программу, которая определяет максимальное число из введённой с клавиатуры последовательности положительных чисел.(длина последовательности неограниченна) 2. Напишите программу, которая проверяет, является ли введённое пользователем целое число простым. (Простое число – это то, которое делится само на себя: 1,3,5,7,11,13,17,19 ит.д.)
4.3	**Задачи на циклы while do** 1. . Написать программу, которая выводит таблицу значений функции **$y=-2x^2-5x-8$** в диапазоне от **–4** до **+4,** с шагом **0,5**
5.1	Написать программы на языке C++ 1. В массив $A[N]$ занесены натуральные числа. Найти сумму элементов, которые кратны данному K. 2. В целочисленной последовательности есть нулевые элемент. Создать массив из номеров этих элементов. 3. Дана последовательность целых чисел $a_1, a_2, ..., a_n$. Выяснить, какое число встречается раньше — положительное или отрицательное. 4. Дана последовательность действительных чисел $a_1, a_2, ..., a_n$. Выяснить, будет ли она возрастающей.

	5. Дана последовательность натуральных чисел a_1, a_2, ..., a_n. Создать массив из четных чисел этой последовательности. Если таких чисел нет, то вывести сообщение об этом факте.
6.1	1 Разобрать текст программы, написанной на языке C++ **# include <stdio.h>** **# include <iostream.h>** **# include <MATH.H>** **# include <STDLIB.H>** **# define n 5** **# define m 17** **# define s 10** **void main()** **{** **int a[n][m];**

```
int q,w,temp=0;
randomize();
// _____________________________________________________++
for (q=0;q<n;q++)for(w=0;w<m;w++)a[q][w]=random(s);
// _____________________________________________________++
for (q=1;q<m;q=q+2)for(w=0;w<n;w++)if (a[w][q]==0)temp++;
// _____________________________________________________++
cout<<"A_\n";
for (q=0;q<n;q++)      {
                        for(w=0;w<m;w++)cout<<a[q][w]<<" ";
                    cout<<"\n";
                        }
cout<<" "<<temp;
cout<<"\n\nPress Alt+F4";
```

	}
6.2	1. Вычисляя значения переменной X=8D+F при всех значениях D=l,2,3 и F=-l,-3,-6. Создать двумерный массив. Вывести значения элементов этого массива и значения D,F. Подсчитать сумму неотрицательных элементов массива. 2. Создать двумерный массив из случайно сгенерированных чисел (от 1 до 36). Подсчитать сумму нечетных чисел элементов массива. Выдать на экран массив и найденную сумму.
6.3	1. Ввести данные для формирования двумерного массива X[4,4], формируя вначале строки. По запросу поменять местами строки или колонки, введя их но мера. 2. Создать трехмерный массив из случайно сгенерированных вещественных чисел, заменяя те элементы массива на 0, которые > 55.0. 3.Ввести вещественные числа для формирования массива A[3,3], подсчитать сумму диагональных элементов этого массива
6.4	1. Ввести вещественные числа для формирования массива A[4,4]. Переставить колонки этого двумерного массива: 1 на место 2 , 2 на место 3, 3 на место 4, 4 на место 1. 2. Ввести данные для формирования двумерного массива L[3,3], формируя в начале строки. Поменять местами строки и колонки: 1 строку перенести в 1 ко лонку, 2 строку перенести во 2 колонку, а 3 строку поместить в 3 колонку.

	3. Сгенерировать двумерный массив. Вычислить сумму элементов обратной диагонали.
7.1	Написать программы на языке C++ 1. Ввести x и вычислить значения всех известных процедур и функций. 2. Нарисовать их графики по заданному примеру(график **SIN(x)**)
8.1	1. Написать функцию, которая вычисляет объём цилиндра. Параметрами функции должны быть радиус и высота цилиндра. 1. Написать функцию, которая сравнивает два целых числа и возвращает результат сравнения в виде одного из знаков > , < или =. 1. Написать функцию, которая вычисляет $\mathbf{a}^{b}$. Числа a и b могут быть дробными положительными числами. 1. Написать функцию Dohod, которая вычисляет доход по вкладу. Исходными данными для функции являются: величина вклада, процентная ставка (годовых) и срок вклада (количество дней)

9.1	Написать программы на языке С++ 1. Подсчитать количество слов в тексте. 2. Заменить одно слово из заданного текста на другое
10.1	1.Считать из файла любую фразу и подсчитать количество слов в ней. 2.Дана последовательность слов, напечатать слова, выполнив преобразования – поменять местами первое и второе слово. Слова считать из файла
10.2	1 Заменить в предложении одно слово на другое, предложение считать из файла 2.Определить, сколько раз в тексте встречается заданная буква, предложение считать из файла. 3.Подсчитать, сколько слов в тексте начинается на заданную букву, предложение считать из файла
11.1	В качестве комплексного примера рассмотрим класс *Point,* который позволяет сформулировать точку на экране компьютера. Поместим описание класса в файл с именем *point.h:*

```
// Файл point.h

#ifndef POINT_H

#define POINT_H 1

class Point                 //класс для определения точки на экране

{

protected:                  //защищенный статус доступа к элементам данных

int x;                      //координата x точки

int y;                      //координата y точки
// Прототипы методов:

public:                     //общедоступный статус доступа

Point  (int,  int);          //конструктор

int putx() t                 //доступ к x

int puty();                 //доступ к y

void show0;                 //изобразить точку на экране
```

	void move (int,int); //переместить точку private: //собственный статус доступа void hide(); //убрать изображение точки }; #endif Поскольку описание класса *Point* планируется использовать при описании других классов, то для предотвращения недопустимого дублирования описания класса в текст включены три директивы препроцессора #define*POINTH, #definePOINT_H 1* и *#endif.* Компонентами класса *Point* являются два элемента данных *x* и *y* с защищенным статусом доступа, пять общедоступных методов и один метод с собственным статусом доступа. Методы в описании класса представлены своими прототипами
12.1	Выполним внешнее описание методов класса, разместив описания в файле Point.cpp **//Файл point.cpp - описание методов**

```
#ifndef POINT_CPP

#define POINT_CPP 1

#include <graphics.h>// прототипы функций графической библиотеки

#include "f:\POS_C\PRIMER\point.h"      // описание класса Point

// Конструктор:

Point  ::  Point  {int xl=0,  int yl=())

{

x=xl;

y=yl;

   }

//Метод доступа  к  x:

int Point  :: putx()

{

return x;
```

```
}
//Метод доступ  к y:
 int Point  :: puty()
{
return y;
}
//Метод  изображения точки на экране:
void Point  ::  show(void)
{
putpixel(x,y,getcolor()) ;
}
// Метод удаления точки с экрана:
void Point  :: hide(void)
{
```

```
putpixel(x,y,getbkcolor());

}

// Метод перемещения точки в новое место экрана:

void Point :: move(int xn=0,  int yn=0)

{

hide();

x=xn;

y=yn;

show();

}

#endif
```

Для получения изображения точки на экране в программу должен быть включен заголовочный файл *graphics.h, B* данном файле находятся прототипы графических функций.

	Достоинство *внешнего описания методов класса* состоит в том, что оно позволяет при необходимости модифицировать содержание методов, причем эти изменения останутся незамеченными для остальных частей программы
13.1	**Задание**Программа, содержащая в своем составе главную функцию, будет выглядеть следующим образом: **// Точка на экране** #include <iostream.h> #include <graphics.h> //прототипы графических функций #include <conio.h> //прототип функции getch() #include "f:\POS_C\PRIMER\point.cpp" //описание класса Point int main() { Point t(100,150); //создана невидимая точка t(x=100,y=150) Point tl(200,200); //создана невидимая точка tl(X=200,Y=200) // Инициализация графики

```
int a=DETECT,b;

initgraph(&a,&b,"F:\\BC5\\bgi");

t.show();       .              / / показать точку t

cout « "\n коор-ты точки t: x=" « t.putx()  « ",y=" « t.puty();
getch();                      //ждать нажатия клавиши

tl.show();                    //показать тоочку tl

cout « "\n координаты точки tl: x=" « tl.putx()  « ",y=" «

tl.puty();

getch();

t.move(150,300);        //переместить точку t(x=150,y=300)

cout « "\n новые координаты t: x=" « t.putx()  « ",y=" «

t.puty(); getch();

closegraph();               //закрыть графический режим

}
```

	В программе используется функция *getch( ),* которая позволяет остановить выполнение программы до нажатия на клавиатуре любой клавиши. Прототип этой функции находится в файле conio.h.
14.1	**// Файл point.h** #ifndef POINT_H #define POINT_H 1 class Point //класс для определения точки на экране { protected: //защищенный статус доступа к элементам данных int x; //координата x точки int y; //координата y точки // Прототипы методов: public: //общедоступный статус доступа Point (int, int); //конструктор int putx() *t* //доступ к x

```
int puty();                 //доступ к y

void show0;                 //изобразить точку на экране

void move  (int,int);    //переместить точку
private:                    //собственный статус доступа
void hide();                //убрать изображение точки

};

#endif
```

Поскольку описание класса *Point* планируется использовать при описании других классов, то для предотвращения недопустимого дублирования описания класса в текст включены три директивы препроцессора #define*POINTH, #definePOINT_H 1* и #*endif.*

Компонентами класса *Point* являются два элемента данных *x* и *y* с защищенным статусом доступа, пять общедоступных методов и один метод с собственным статусом доступа. Методы в описании класса представлены своими прототипами

14.2 Выполним внешнее описание методов класса, разместив описания в файле

Point.cpp

//Файл point.cpp - описание методов

```
#ifndef POINT_CPP
#define POINT_CPP 1
#include <graphics.h>// прототипы функций графической библиотеки
#include "f:\POS_C\PRIMER\point.h"      // описание класса Point
// Конструктор:
Point  ::  Point  {int xl=0,  int yl=())
{
x=xl;
y=yl;
   }
//Метод доступа  к  x:
int Point  :: putx()
```

```
{
return x;
}
//Метод доступ  к y:
 int Point  :: puty()
{
return y;
}
//Метод  изображения точки на экране:
void Point  ::  show(void)
{
putpixel(x,y,getcolor()) ;
}
// Метод удаления точки с экрана:
```

```
void Point  :: hide(void)

{

putpixel(x,y,getbkcolor());

}

// Метод перемещения точки в новое место экрана:

void Point :: move(int xn=0,  int yn=0)

{

hide();

x=xn;

y=yn;

show();

}

#endif
```

	Для получения изображения точки на экране в программу должен быть включен заголовочный файл *graphics.h, В* данном файле находятся прототипы графических функций. ***Достоинство*** *внешнего описания методов класса* состоит в том, что оно позволяет при необходимости модифицировать содержание методов, причем эти изменения останутся незамеченными для остальных частей программы
14.3	**Задание**Программа, содержащая в своем составе главную функцию, будет выглядеть следующим образом: **// Точка на экране** #include <iostream.h> #include <graphics.h> //прототипы графических функций #include <conio.h> //прототип функции getch() #include "f:\POS_C\PRIMER\point.cpp" //описание класса Point int main() { Point t(100,150); //создана невидимая точка t(x=100,y=150) Point tl(200,200); //создана невидимая точка tl(X=200,Y=200)

```
// Инициализация графики

int a=DETECT,b;

initgraph(&a,&b,"F:\\BC5\\bgi");

t.show();       .              / / показать точку t

cout « "\n коор-ты точки t: x=" « t.putx()  « ",y=" « t.puty();
getch();                      //ждать нажатия клавиши

tl.show();                    //показать тоочку tl

cout « "\n координаты точки tl: x=" « tl.putx()  « ",y=" «

tl.puty();

getch();

t.move(150,300);        //переместить точку t(x=150,y=300)

cout « "\n новые координаты t: x=" « t.putx()  « ",y=" «

t.puty(); getch();

closegraph();               //закрыть графический режим
```

	} В программе используется функция *getch(),* которая позволяет остановить выполнение программы до нажатия на клавиатуре любой клавиши. Прототип этой функции находится в файле conio.h.
15.1	В качестве комплексного примера рассмотрим класс *Point,* который позволяет сформулировать точку на экране компьютера. Поместим описание класса в файл с именем *point.h:* **// Файл point.h** #ifndef POINT_H #define POINT_H 1 class Point //класс для определения точки на экране { protected: //защищенный статус доступа к элементам данных int x; //координата x точки

```
int y;                    //координата y точки
// Прототипы методов:

public:                   //общедоступный статус доступа

Point  (int,  int);       //конструктор

int putx() t              //доступ к x

int puty();               //доступ к y

void show0;               //изобразить точку на экране

void move  (int,int);     //переместить точку
private:                  //собственный статус доступа
void hide();              //убрать изображение точки

};

#endif
```

Поскольку описание класса *Point* планируется использовать при описании других классов, то для предотвращения недопустимого дублирования описания класса в текст включены три директивы препроцессора #define*POINTH, #definePOINT_H 1* и #*endif.*

	Компонентами класса *Point* являются два элемента данных *x* и *y* с защищенным статусом доступа, пять общедоступных методов и один метод с собственным статусом доступа. Методы в описании класса представлены своими прототипами
15.2	Выполним внешнее описание методов класса, разместив описания в файле Point.cpp **//Файл point.cpp - описание методов** #ifndef POINT_CPP #define POINT_CPP 1 #include <graphics.h>// прототипы функций графической библиотеки #include "f:\POS_C\PRIMER\point.h" // описание класса Point // Конструктор: Point :: Point {int xl=0, int yl=()) {

```
x=xl;
y=yl;
   }
//Метод доступа  к  x:
int Point  :: putx()
{
return x;
}
//Метод доступ  к y:
 int Point  :: puty()
{
return y;
}
//Метод  изображения точки на экране:
```

```
void Point  ::  show(void)

{

putpixel(x,y,getcolor()) ;

}

// Метод удаления точки с экрана:

void Point  :: hide(void)

{

putpixel(x,y,getbkcolor());

}

// Метод перемещения точки в новое место экрана:

void Point :: move(int xn=0,  int yn=0)

{

hide();

x=xn;
```

	y=yn; show(); } #endif Для получения изображения точки на экране в программу должен быть включен заголовочный файл *graphics.h, B* данном файле находятся прототипы графических функций. ***Достоинство*** *внешнего описания методов класса* состоит в том, что оно позволяет при необходимости модифицировать содержание методов, причем эти изменения останутся незамеченными для остальных частей программы
15.3	**Задание**Программа, содержащая в своем составе главную функцию, будет выглядеть следующим образом: **// Точка на экране** #include <iostream.h> #include <graphics.h> //прототипы графических функций #include <conio.h> //прототип функции getch() #include "f:\POS_C\PRIMER\point.cpp" //описание класса Point

```
int main()

{

Point  t(100,150);       //создана невидимая точка t(x=100,y=150)

Point tl(200,200);       //создана невидимая точка tl(X=200,Y=200)

// Инициализация графики

int a=DETECT,b;

initgraph(&a,&b,"F:\\BC5\\bgi");

t.show();       .              / / показать точку t

cout « "\n коор-ты точки t: x=" « t.putx()  « ",y=" « t.puty();
getch();                      //ждать нажатия клавиши

tl.show();                    //показать тоочку tl

cout « "\n координаты точки tl: x=" « tl.putx()  « ",y=" «

tl.puty();

getch();
```

```
t.move(150,300);        //переместить точку t(x=150,y=300)

cout « "\n новые координаты t: x=" « t.putx()  « ",y=" «

t.puty(); getch();

closegraph();           //закрыть графический режим

}
```

В программе используется функция *getch(),* которая позволяет остановить выполнение программы до нажатия на клавиатуре любой клавиши. Прототип этой функции находится в файле conio.h.

16.1 В качестве комплексного примера рассмотрим класс *Point,* который позволяет сформулировать точку на экране компьютера. Поместим описание класса в файл с именем *point.h:*

```
// Файл point.h

#ifndef POINT_H

#define POINT_H 1

class Point             //класс для определения точки на экране
```

```
{

protected:                //защищенный статус доступа к элементам данных

int x;                    //координата x точки

int y;                    //координата y точки
// Прототипы методов:

public:                   //общедоступный статус доступа

Point  (int,  int);       //конструктор

int putx() t              //доступ к x

int puty();               //доступ к y

void show0;               //изобразить точку на экране

void move  (int,int);     //переместить точку
private:                  //собственный статус доступа
void hide();              //убрать изображение точки

};

#endif
```

	Поскольку описание класса *Point* планируется использовать при описании других классов, то для предотвращения недопустимого дублирования описания класса в текст включены три директивы препроцессора #define*POINTH, #definePOINT_H 1* и #*endif.* Компонентами класса *Point* являются два элемента данных *x* и *y* с защищенным статусом доступа, пять общедоступных методов и один метод с собственным статусом доступа. Методы в описании класса представлены своими прототипами
16.2	Выполним внешнее описание методов класса, разместив описания в файле Point.cpp **//Файл point.cpp - описание методов** #ifndef POINT_CPP #define POINT_CPP 1 #include <graphics.h>// прототипы функций графической библиотеки #include "f:\POS_C\PRIMER\point.h" // описание класса Point // Конструктор:

```
Point  ::  Point  {int xl=0,  int yl=())

{

x=xl;

y=yl;

   }

//Метод доступа  к  x:

int Point  :: putx()

{

return x;

}

//Метод доступ  к y:

 int Point  :: puty()

{

return y;
```

```
}
//Метод  изображения точки на экране:
void Point  ::  show(void)
{
putpixel(x,y,getcolor()) ;
}
// Метод удаления точки с экрана:
void Point  :: hide(void)
{
putpixel(x,y,getbkcolor());
}
// Метод перемещения точки в новое место экрана:
void Point :: move(int xn=0,  int yn=0)
{
```

	hide(); x=xn; y=yn; show(); } #endif Для получения изображения точки на экране в программу должен быть включен заголовочный файл *graphics.h, B* данном файле находятся прототипы графических функций. ***Достоинство*** *внешнего описания методов класса* состоит в том, что оно позволяет при необходимости модифицировать содержание методов, причем эти изменения останутся незамеченными для остальных частей программы
16.3	**Задание**Программа, содержащая в своем составе главную функцию, будет выглядеть следующим образом: **// Точка на экране** #include <iostream.h> #include <graphics.h> //прототипы графических функций

```
#include <conio.h>      //прототип функции getch()

#include "f:\POS_C\PRIMER\point.cpp"   //описание класса Point

int main()

{

Point  t(100,150);       //создана невидимая точка t(x=100,y=150)

Point tl(200,200);       //создана невидимая точка tl(X=200,Y=200)

// Инициализация графики

int a=DETECT,b;

initgraph(&a,&b,"F:\\BC5\\bgi");

t.show();       .              / / показать точку t

cout « "\n коор-ты точки t: x=" « t.putx()  « ",y=" « t.puty();
getch();                      //ждать нажатия клавиши

tl.show();                    //показать тоочку tl

cout « "\n координаты точки tl: x=" « tl.putx()  « ",y=" «
```

	tl.puty(); getch(); t.move(150,300); //переместить точку t(x=150,y=300) cout « "\n новые координаты t: x=" « t.putx() « ",y=" « t.puty(); getch(); closegraph(); //закрыть графический режим } В программе используется функция *getch(),* которая позволяет остановить выполнение программы до нажатия на клавиатуре любой клавиши. Прототип этой функции находится в файле conio.h.
17.1	Создание различных тестов в среде Delphi
18.1	Создание WEB - страниц с использование языка гиперссылок HTML и JAVA Script

Перечень показателей текущего контроля

Номер п	Значение показателя

оказателя	
1.1.1.1	Умение работать в среде C++
1.1.1.2	Умение использовать типы данных при разработке программ
1.2.1.1	Умение использовать типы данных при разработке программ
2.1.1.1	Знание операторов условия и выбора
2.1.2.1	Знание операторов условия и выбора
2.2.1.1	Умение работать с операторами условия и операторами выбора
3.1.1.1	Знание операторов цикла
3.1.2.1	Знание операторов цикла
3.2.1.1	Умение использовать в программах операторы цикла
4.1.1.1	Знание операторов цикла (Do While и While do)
4.1.2.1	Знание операторов цикла (Do While и While do)
4.2.1.1	Знание операторов цикла (Do While и While do)
5.1.1.1	Знание работы с одномерными массивами
5.1.2.1	Знание работы с одномерными массивами
5.2.1.1	Умение работы с одномерными массивами
6.1.1.1	Знание работы с двухмерными массивами
6.1.2.1	Знание работы с двухмерными массивами
6.1.3.1	Знание работы с двухмерными массивами
6.2.1.1	Умение работы с двухмерными массивами
6.2.2.1	Умение работы с двухмерными массивами

6.2.3.1	Умение работы с двухмерными массивами
6.2.4.1	Умение работы с двухмерными массивами
6.2.5.1	Умение работы с двухмерными массивами
7.1.1.1	Знание работы со стандартными функциями
7.1.2.1	Знание работы со стандартными функциями
7.1.3.1	Знание работы со стандартными функциями
7.2.1.1	Умение работы со стандартными функциями
8.1.1.1	Знание использования в программах процедур и функций пользователя.
8.1.2.1	Знание использования в программах процедур и функций пользователя
8.1.3.1	Знание использования в программах процедур и функций пользователя
8.2.1.1	Умение использования в программах процедур и функций пользователя
9.1.1.1	Знание строковых процедур
9.1.2.1	Знание строковых процедур
9.1.3.1	Знание строковых процедур
9.2.1.1	Умение использовать при программировании строковых процедур
10.1.1.1	Знание работы с файлами
10.1.2.1	Знание работы с файлами
10.1.3.1	Знание работы с файлами
10.2.1.1	Умение работы с файлами
11.1.1.1	Знание разработки прорамм с использованием классов

11.1.2.1	Знание разработки прорамм с использованием классов
11.1.3.1	Знание разработки прорамм с использованием классов
11.2.1.1	Знание разработки прорамм с использованием классов
11.2.2.1	Умение разработки прорамм с использованием классов
12.1.1.1	Знание Классов и наследование.
12.1.2.1	Знание Классов и наследование
12.1.3.1	Знание Классов и наследование
12.2.1.1	Знание Классов и наследование
12.2.2.1	Знание Классов и наследование
13.1.1.1	Знание классов, наследования, полиморфизм
13.1.2.1	Знание классов, наследования, полиморфизм
13.1.3.1	Знание классов, наследования, полиморфизм
13.2.1.1	Знание классов, наследования, полиморфизм
14.1.1.1	Знание классов, наследования,полиморфизма,исключений
14.1.2.1	Знание классов, наследования,полиморфизма,исключений
14.1.3.1	Знание классов, наследования,полиморфизма,исключений
14.2.1.1	Знание классов, наследования,полиморфизма,исключений
15.1.2.1	Знание классов, наследования,полиморфизма, исключений
15.2.1.1	Знание классов, наследования,полиморфизма, исключений
16.1.2.1	Знание классов, наследования, полиморфизм, исключения, потоковый ввод-вывод
16.2.1.1	Знание классов, наследования, полиморфизм, исключения, потоковый ввод-вывод
16.2.2.1	Знание классов, наследования, полиморфизм, исключения, потоковый ввод-вывод

17.1.2.1	Осмысление выбранной темы курсовой работы, сравнение с аналогами.
17.1.3.1	Осмысление выбранной темы курсовой работы, сравнение с аналогами.
17.1.4.1	Осмысление выбранной темы курсовой работы, сравнение с аналогами.
17.2.1.1	Осмысление выбранной темы курсовой работы, сравнение с аналогами.
17.2.2.1	Осмысление выбранной темы курсовой работы, сравнение с аналогами.
17.2.3.1	Осмысление выбранной темы курсовой работы, сравнение с аналогами.
17.2.4.1	Умение работать с методическими указаниями к курсовой работе
17.2.5.1	Умение работать с методическими указаниями к курсовой работе
18.1.2.1	Осмысление выбранной темы курсовой работы, сравнение с аналогами.
18.1.3.1	Осмысление выбранной темы курсовой работы, сравнение с аналогами.
18.1.4.1	Знание методических указаний к оформлению курсовой работы
18.2.1.1	Осмысление выбранной темы курсовой работы, сравнение с аналогами.
18.2.2.1	Осмысление выбранной темы курсовой работы, сравнение с аналогами.
18.2.3.1	Осмысление выбранной темы курсовой работы, сравнение с аналогами.
18.2.4.1	Умение работать с методическими указаниями к оформлению курсовой работы
18.2.5.1	Умение работать с методическими указаниями к оформлению курсовой работы

3. РЕЗУЛЬТАТЫ ОСВОЕНИЯ УЧЕБНОЙ ПРАКТИКИ, ПОДЛЕЖАЩИЕ ПРОВЕРКЕ НА ТЕКУЩЕМ КОНТРОЛЕ

Учебная практика направлена на формирование у обучающихся практических профессиональных умений, приобретение первоначального практического опыта, реализуется в рамках профессионального модуля по основному виду профессиональной деятельности для последующего освоения ими общих и профессиональных компетенций по избранной специальности. Предметом оценки по учебной практике являются дидактические

единицы:
По учебной практике обучающиеся ведут дневник практики, в котором выполняют записи о решении профессиональных задач, выполнении заданий в соответствии с программой, ежедневно подписывают дневник с отметкой о выполненных работах у руководителя практики. Оценка по учебной практике выставляется на основании аттестационного листа.

3.1 Фонд оценочных средств учебной практики

Текущий контроль №1

Индекс тем МДК модуля	Наименование вида работ (на которой осуществляется текущий контроль)	Индекс дидактической единицы	Индекс формируемой компетенции	Основные показатели оценивания результата	№ задания относящийся к показателю оценивания	Метод контроля	Форма контроля	Вид контроля
1.1.2	Создание программ с использованием команд сдвигов	2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.1.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	2.2.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

			ПК.1.6				
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.3.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.4.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.5	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.5.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.1.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.2	ПК.1.1,	3.2.1	1, 2	Сравнение с	Практическая	Защита

			ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6			аналогом	работа	
		3.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.3.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.4.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

Перечень заданий текущего контроля

Номер задания	Задания
1	**Задание** Создать com-файл. Расписать в тетради, как происходит сдвиг битов в байтах при разных сдвигах**.** **CCC SEGMENT**

```
        assume      DS:CCC,CS:CCC,SS:CCC
        ORG         100H
VX:     JMP         PP
FIF     DB          5
PP      PROC        NEAR
        MOV         AL,FIF
        SHR         AL,1
        MOV         AL,-5
        SHR         AL,1

        MOV         AL,FIF
        SHL         AL,1
        MOV         AL,-5
        SHL         AL,1
```

```
        MOV     AL,FIF
        SAR     AL,1
        MOV     AL,-5
        SAR     AL,1

        MOV     AL,FIF
        SAL     AL,1
        MOV     AL,-5
        SAL     AL,1

        RET
pp      ENDP
ccc     ENDS
```

	END VX
2	**Задание** Создать COM-файл **title Primer1** **add segment para** **assume ss:add,ds:add,cs:add** **org 100h** **vv: jmp nach** **nam1 db '123456'** **nam2 db 'abcdef'** **nach proc near** **lea si,nam1** **lea di,nam2**

```
        mov     cx,6
m1:
        mov     al,[si]
        mov     ah,[di]
        mov     [si],ah
        mov     [di],al
        inc     si
        inc     di
        dec     cx
        jnz     m1
        ret
nach    endp
add     ends
        end vv
```

Перечень показателей текущего контроля

Номер показателя	Значение показателя
2.1.1	Умение программировать циклические алгоритмы на языке Ассемблер
2.2.1	Умение программировать циклические алгоритмы на языке Ассемблер
2.3.1	Умение программировать циклические алгоритмы на языке Ассемблер
2.4.1	Умение программировать циклические алгоритмы на языке Ассемблер
2.5.1	Умение программировать циклические алгоритмы на языке Ассемблер
3.1.1	Умение программировать циклические алгоритмы на языке Ассемблер
3.2.1	Умение программировать циклические алгоритмы на языке Ассемблер
3.3.1	Умение программировать циклические алгоритмы на языке Ассемблер
3.4.1	Умение программировать циклические алгоритмы на языке Ассемблер

Текущий контроль №2

Индекс тем МДК модуля	Наименование вида работ (на которой осуществляется текущий контроль)	Индекс дидактической единицы	Индекс формируемой компетенции	Основные показатели оценивания результата	№ задания относящийся к показателю оценивания	Метод контроля	Форма контроля	Вид контроля
1.1.2	Составление	2.1	ПК.1.1,	2.1.1	1, 2	Сравнение с	Практическая	Защита

	программы с использованием ввода-вывода на экран		ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6			аналогом	работа	
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.2.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.3.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.4.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.5	ПК.1.1, ПК.1.2, ПК.1.3,	2.5.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

			ПК.1.4, ПК.1.5, ПК.1.6					
		3.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.1.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.2.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.3.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	3.4.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

			ПК.1.6					

Перечень заданий текущего контроля

Номер задания	Задания
1	**Задание** Написать программу на языке Ассемблер, выбрав одну из предложенных задач: 1.Ввести строку символьных данных ,задавая буфер равный 18 байт. Подсчитать в этой строке количество символов "i" . Выдать подсчитанное количество символов.
2	2.Ввести строку символьных данных ,задавая буфер равный 10 байт. Проанализировать встречающиеся символы . Выдать одно из сообщений: "Символы русского регистра" "Символы латинского регистра" "Символы и русского и латинского регистров".

Перечень показателей текущего контроля

Номер показателя	Значение показателя
2.1.1	Умение программировать используя подпрограммы ввода-вывода
2.2.1	Умение программировать используя подпрограммы ввода-вывода
2.3.1	Умение программировать используя подпрограммы ввода-вывода
2.4.1	Умение программировать используя подпрограммы ввода-вывода
2.5.1	Умение программировать используя подпрограммы ввода-вывода
3.1.1	Умение программировать используя подпрограммы ввода-вывода
3.2.1	Умение программировать используя подпрограммы ввода-вывода
3.3.1	Умение программировать используя подпрограммы ввода-вывода
3.4.1	Умение программировать используя подпрограммы ввода-вывода

Текущий контроль №3

Индекс тем МДК модуля	Наименование вида работ (на которой осуществляется текущий контроль)	Индекс дидактической единицы	Индекс формируемой компетенции	Основные показатели оценивания результата	№ задания относящийся к показателю оценивания	Метод контроля	Форма контроля	Вид контроля
2.1.1	Выполнение программ на работу с двухмерными	2.1	ПК.1.1, ПК.1.2, ПК.1.3,	2.1.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

	массивами в среде Borland C++		ПК.1.4, ПК.1.5, ПК.1.6					
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.2.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.3.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.4.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.5	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	2.5.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

			ПК.1.6					
		3.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.1.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.2.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.3.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.4.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

Перечень заданий текущего контроля

Номер задания	Задания
1	1.Написать программу, которая выводит таблицу квадратов десяти первых положительных чисел Написать программу, которая вычисляет сумму первых членов ряда: 1,3,5,7,……Количество суммируемых членов ряда задаётся во время работы программы.
2	Написать программу, которая выводит таблицу степеней двойки от нулевой до десятой. 4. Написать программу, которая выводит таблицу значений функции **y=-2,4x ^{2} +5x-3** в диапазоне от **–2** до **+2,** с шагом **0,5**

Перечень показателей текущего контроля

Номер п	Значение показателя

оказателя	
2.1.1	Умение программировать циклические алгоритмы в среде C++ Builder
2.2.1	Умение программировать циклические алгоритмы в среде C++ Builder
2.3.1	Умение программировать циклические алгоритмы в среде C++ Builder
2.4.1	Умение программировать циклические алгоритмы в среде C++ Builder
2.5.1	Умение программировать циклические алгоритмы в среде C++ Builder
3.1.1	Умение программировать циклические алгоритмы в среде C++ Builder
3.2.1	Умение программировать циклические алгоритмы в среде C++ Builder
3.3.1	Умение программировать циклические алгоритмы в среде C++ Builder
3.4.1	Умение программировать циклические алгоритмы в среде C++ Builder

Текущий контроль №4

Индекс тем МДК модуля	Наименование вида работ (на которой осуществляется текущий контроль)	Индекс дидактической единицы	Индекс формируемой компетенции	Основные показатели оценивания результата	№ задания относящийся к показателю оценивания	Метод контроля	Форма контроля	Вид контроля
2.1.1	. Выполнение программ на сортировки	2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4,	2.1.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

			ПК.1.5, ПК.1.6				
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.2.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.3.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.4.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.5	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.5.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

		3.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.1.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.2.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.3.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.4.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

Перечень заданий текущего контроля

Номер задания	Задания
1	Написать программы на языке C++ 1. В массив $A[N]$ занесены натуральные числа. Найти сумму элементов, которые кратны данному K. 2. В целочисленной последовательности есть нулевые элемент. Создать массив из номеров этих элементов.
2	1. Дана последовательность целых чисел a_1, a_2, ..., a_n. Выяснить, какое число встречается раньше — положительное или отрицательное. 2. Дана последовательность действительных чисел a_1, a_2, ..., a_n. Выяснить, будет ли она возрастающей. Дана последовательность натуральных чисел a_1, a_2, ..., a_n. Создать массив из четных чисел этой последовательности. Если таких чисел нет, то вывести сообщение об этом факте

Перечень показателей текущего контроля

Номер показателя	Значение показателя
2.1.1	Умение программировать сортировки, используя разные алгоритмы
2.2.1	Умение программировать сортировки, используя разные алгоритмы
2.3.1	Умение программировать сортировки, используя разные алгоритмы
2.4.1	Умение программировать сортировки, используя разные алгоритмы

2.5.1	Умение программировать сортировки, используя разные алгоритмы
3.1.1	Умение программировать сортировки, используя разные алгоритмы
3.2.1	Умение программировать сортировки, используя разные алгоритмы
3.3.1	Умение программировать сортировки, используя разные алгоритмы
3.4.1	Умение программировать сортировки, используя разные алгоритмы

Текущий контроль №5

Индекс тем МДК модуля	Наименование вида работ (на которой осуществляется текущий контроль)	Индекс дидактической единицы	Индекс формируемой компетенции	Основные показатели оценивания результата	№ задания относящийся к показателю оценивания	Метод контроля	Форма контроля	Вид контроля
2.1.2	Объектно-ориентированное программирование. Примеры на наследование	2.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.1.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.2	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5,	2.2.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

			ПК.1.6				
		2.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.3.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.4.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		2.5	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	2.5.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.1	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.1.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.2	ПК.1.1,	3.2.1	1, 2	Сравнение с	Практическая	Защита

			ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6			аналогом	работа	
		3.3	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.3.1	1, 2	Сравнение с аналогом	Практическая работа	Защита
		3.4	ПК.1.1, ПК.1.2, ПК.1.3, ПК.1.4, ПК.1.5, ПК.1.6	3.4.1	1, 2	Сравнение с аналогом	Практическая работа	Защита

Перечень заданий текущего контроля

Номер задания	Задания
1	**Задание** В качестве комплексного примера рассмотрим класс *Point,* который позволяет сформулировать точку на экране компьютера. Поместим описание класса в файл с именем *point.h:*

```
// Файл point.h

#ifndef POINT_H

#define POINT_H 1

class Point                //класс для определения точки на экране

{

protected:                 //защищенный статус доступа к элементам данных

int x;                     //координата x точки

int y;                     //координата y точки
// Прототипы методов:

public:                    //общедоступный статус доступа

Point  (int,  int);         //конструктор

int putx() t                //доступ к x

int puty();                //доступ к y

void show0;                //изобразить точку на экране
```

	void move (int,int); //переместить точку private: //собственный статус доступа void hide(); //убрать изображение точки }; #endif
2	**Задание** Выполним внешнее описание методов класса, разместив описания в файле Point.cpp **//Файл point.cpp - описание методов** #ifndef POINT_CPP #define POINT_CPP 1 #include <graphics.h>// прототипы функций графической библиотеки #include "f:\POS_C\PRIMER\point.h" // описание класса Point // Конструктор:

```
Point  ::  Point  {int xl=0,  int yl=())

{

x=xl;

y=yl;

   }

//Метод доступа  к  x:

int Point  :: putx()

{

return x;

}

//Метод доступ  к y:

 int Point  :: puty()

{

return y;
```

```
}
//Метод  изображения точки на экране:
void Point  ::  show(void)
{
putpixel(x,y,getcolor()) ;
}
// Метод удаления точки с экрана:
void Point  :: hide(void)
{
putpixel(x,y,getbkcolor());
}
// Метод перемещения точки в новое место экрана:
void Point :: move(int xn=0,  int yn=0)
{
```

	hide(); x=xn; y=yn; show(); } #endif

Перечень показателей текущего контроля

Номер показателя	Значение показателя
2.1.1	Умение программировать, используя ООП (объекты, классы, конструкторы, деструкторы, наследование)
2.2.1	Умение программировать, используя ООП (объекты, классы, конструкторы, деструкторы, наследование)
2.3.1	Умение программировать, используя ООП (объекты, классы, конструкторы, деструкторы, наследование)
2.4.1	Умение программировать, используя ООП (объекты, классы, конструкторы, деструкторы, наследование)
2.5.1	Умение программировать, используя ООП (объекты, классы, конструкторы, деструкторы, наследование)
3.1.1	Умение программировать, используя ООП (объекты, классы, конструкторы, деструкторы, наследование)

3.2.1	Умение программировать, используя ООП (объекты, классы, конструкторы, деструкторы, наследование)
3.3.1	Умение программировать, используя ООП (объекты, классы, конструкторы, деструкторы, наследование)
3.4.1	Умение программировать, используя ООП (объекты, классы, конструкторы, деструкторы, наследование)

4. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ, ИСПОЛЬЗУЕМЫЙ ДЛЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

МДК.01.01 Системное программирование, семестр 6, Дифференцированный зачет

Автоматический контроль по результатам текущего контроля
Текущий контроль №1
Текущий контроль №2
Текущий контроль №3
Текущий контроль №4
Текущий контроль №5
Текущий контроль №6
Текущий контроль №7
Текущий контроль №8
Текущий контроль №9
Текущий контроль №10
Текущий контроль №11
Текущий контроль №12

МДК.01.02 Прикладное программирование, семестр 6, Дифференцированный зачет

Автоматический контроль по результатам текущего контроля
Текущий контроль №14
Текущий контроль №18