Министерство науки и высшего образования Российской Федерации
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)

Институт прикладной математики и компьютерных наук

УТВЕРЖДЕНО:
Директор
А. В. Замятин

Оценочные материалы по дисциплине

Языки программирования

по направлению подготовки / специальности

10.05.01 Компьютерная безопасность

Направленность (профиль) подготовки/ специализация:
Анализ безопасности компьютерных систем

Форма обучения
Очная

Квалификация
Специалист по защите информации

Год приема
2024

СОГЛАСОВАНО:
Руководитель ОП
В.Н. Тренькаев

Председатель УМК
С.П. Сущенко

Томск – 2024

1. Компетенции и индикаторы их достижения, проверяемые данными оценочными материалами

Целью освоения дисциплины является формирование следующих компетенций:

ОПК-7 Способен создавать программы на языках высокого и низкого уровня, применять методы и инструментальные средства программирования для решения профессиональных задач, осуществлять обоснованный выбор инструментария программирования и способов организации программ.

Результатами освоения дисциплины являются следующие индикаторы достижения компетенций:

ИОПК-7.1 Осуществляет построение алгоритма, проведение его анализа и реализации в современных программных комплексах

ИОПК-7.2 Понимает общие принципы построения и использования языков программирования высокого уровня и низкого уровня

ИОПК-7.3 Демонстрирует навыки создания программ с применением методов и инструментальных средств программирования для решения различных профессиональных, исследовательских и прикладных задач

ИОПК-7.4 Осуществляет обоснованный выбор инструментария программирования и способов организации программ

2. Оценочные материалы текущего контроля и критерии оценивания

Третий семестр

Элементы текущего контроля:
– коллоквиумы;
– практические задания, выполняемые на лабораторных занятиях.

Типовые задания для проведения текущего контроля успеваемости по дисциплине (ОПК-7, ИОПК-7.1, ИОПК-7.2, ИОПК-7.3, ИОПК-7.4)

Текущий контроль усвоения теоретического материала проводится в форме коллоквиума. Предполагается проведение двух коллоквиумов. На каждом коллоквиуме студенту предлагается ответить на теоретический вопрос и выполнить практическое задание.

Вопросы к первому коллоквиуму

А) Теоретическая часть.

1. Принципы объектно-ориентированного программирования, их суть. Определение класса. Объект. Сокрытие информации. Чем член-функции отличаются от обычных. Какие существуют типы доступа и чем они отличаются друг от друга. Неявный указатель this.
2. Чем член-функции отличаются от обычных. Операция ::. Чем отличаются член-функции, определенные внутри класса, от функций, определенных вне класса.
3. Конструкторы и деструкторы. Назначение конструктора. Особенности конструктора. Деструктор. Особенности деструктора.
4. Конструктор копирования. В каком случае необходим конструктор копирования и почему. Чем конструктор копирования отличается от перегрузки операции =.
5. Перегрузка операций. Перечислите правила перегрузки операций. Перегрузки операций + и +=. Чем отличается операция + от операции +=.
6. Перегрузка операций. Перечислите правила перегрузки операций. Перегрузка операций [ ], ( ).
7. Перегрузка операций. Перечислите правила перегрузки операций. Особенности перегрузки операции (тип). Перегрузка операции ++.

8. Перегрузка операций. Перечислите правила перегрузки операций. Перегрузка операции =. Когда необходимо перегружать операцию =. Чем конструктор копирования отличается от перегрузки операции =.
9. Дружественность. Что может быть другом класса. Перегрузка операций потокового ввода >> и вывода <<. Что общего и в чем разница между перегрузкой операции потокового вывода << и перегрузкой операции потокового ввода >>. Можно ли перегрузить операции потокового ввода/вывода как методы класса, ответ аргументировать. Обязательно ли использовать ссылку при перегрузке потоковых операций.
10. Массивы объектов. Какие конструкторы можно использовать для инициализации статических объектов. Можно ли явно инициализировать массив объектов, определенных в динамической памяти. Как работает конструктор для массива объектов. Как работает деструктор для массива объектов.
11. Агрегированные классы. Каким образом можно определить конструктор агрегированного класса, если член-данное в нем – указатель на объект другого (используемого) класса? Каким образом агрегированный класс может использовать член-данные используемого класса из части private.
12. Агрегированные классы. Каким образом агрегированный класс может использовать член-данные используемого класса из части private.
13. Функции- и классы-шаблоны. Что такое порожденная функция. Использование функций-шаблонов и классов-шаблонов. Описать работу компилятора.
14. Статические член-данные класса: как объявляется статическое член-данное в классе, его отличие от остальных член-данных; как инициализируется статическое член-данное, как влияет тип доступа на инициализацию статического член-данного; можно ли изменять статическое член-данное в main; как можно получить доступ к статическому член-данному в main.
15. Статические методы класса: как объявляются статические методы в классе, их назначение; как определяются статические методы вне класса; особенности статических методов класса

Б) Практическая часть.

1. Класс Circle (Круг) .
   Член-данные: double x, y (координаты центра) и double r (радиус).
   Методы: все конструкторы (если не требуется конструктора копирования – объяснить почему); ввод и вывод; нахождение площади; проверка, попадает ли заданная точка в круг.
   Перегрузка операторов: операции + и += (умножение круга на число – центр не меняется, радиус умножается на число); сравнение (== и !=; два круга равны, если равны их радиусы); потокового ввода >> и потокового вывода <<.
2. Класс FreeVector (Свободный вектор) .
   Член-данные: double x, y, z (координаты вектора).
   Методы: все конструкторы (если не требуется конструктора копирования – объяснить почему); ввод и вывод; нахождение длины вектора.
   Перегрузка операторов: операции + , += (сумма векторов); операции – , –= (разность векторов); операции * (скалярное произведение векторов); сравнения (== и !=); потокового ввода >> и потокового вывода <<.
3. Класс Complex.
   Член-данные: double x, y (действительная и мнимая часть комплексного числа).
   Методы: все конструкторы (если не требуется конструктора копирования – объяснить почему); ввод, вывод, модуль;комплексно-сопряженное;
   Перегрузка операторов: арифметические операции (+, +=, –, – =, *, *=, /, /=); сравнение (== и !=); потокового ввода >> и потокового вывода <<.
4. Класс массив Array.

Член-данные (2 варианта):
А) указатель на целое, количество взятой памяти (mem), число элементов массива (n);
Б) указатель на целое, число элементов массива (памяти берётся столько же) (n);
Методы:
Для А – Array(int mem_=1, int n_=0) (можно добавить параметр – способ заполнения) , для Б – Array(int n_=0) (можно добавить параметр – способ заполнения);
Array(int*, int) (создание объекта класса Array из целочисленного массива); Array(const Array&); ~Array(); функции ввода/вывода; формирование случайного массива; функция поиска (на входе – ключ, на выходе – индекс, если ключ присутствует в массиве, –1, если ключа в массиве нет); функция нахождения max/min элемента (возвращает индекс элемента); добавление ключа на позицию;
удаление позиции (изменяется существующий объект, создается новый объект).
Перегрузка операторов:
= (присвоение); [ ] (возвращает ссылку); +(int key) – добавление ключа key в конец массива; +=(int key) – добавление ключа key в конец массива; +(Array) – добавление массива в конец массива; +=(Array) – добавление массива в конец массива; –(int key) – удаление ключа key (первое вхождение); –=(int key) – удаление ключа key (первое вхождение; ==, !=(сравнение); потоковый ввод/вывод;

5. Класс String
   Член-данные: char *Str (строка), int Len (размер строки Str).
   Методы: String(char *); String(int); String(const String&); ~String().
   Перегрузка операторов: = (присвоение); [ ] (возвращает ссылку); >> (потоковый ввод); << (потоковый вывод); +, += (конкатенация); == (сравнение строк).
6. Класс-шаблон Stack на основе массива.
   Член-данные: массив, размер массива, реальное число элементов стека.
   Методы: Stack(int k=50); Stack(const Stack<T> &); ~ Stack(); проверка на пустоту; проверка на заполненность; добавить в стек элемент; удалить из стека элемент; взять значение с вершины стека;
   Перегрузка операторов: = (присвоение); потоковый вывод << .

Вопросы ко второму коллоквиуму

А) Теоретическая часть.

1. Базовый и порожденный классы. Тип доступа protected. Типы наследования public, protected и private. Ограничения наследования.
2. Простое наследование. Принцип доминирования в иерархии наследования.
3. Конструктор порожденного класса. Его вид. Деструктор порожденного класса.
4. Стандартные преобразования при наследовании.
5. Множественное наследование. Прямые и не прямые базовые классы. Виртуальный базовый класс. Особенности инициализации его ч/данных.
6. Полиморфизм. Раннее и позднее связывание.
7. Виртуальные функции. Чистые виртуальные функции и абстрактный базовый класс.
8. Правила определения виртуальных функций: описание, определение, виртуальные функции порожденного класса.
9. Правила определения виртуальных функций: виртуальные и невиртуальные вызовы.
10. Правила определения виртуальных функций: виртуальные операции, приватные функции.
11. Механизм позднего связывания.
12. Библиотека STL – БСШ. Определение контейнера. Структура БСШ.
13. Библиотека стандартных шаблонов. Шаблоны vector, list, set, stack, queue
14. Класс «Факультет» (классы Person, Student, Teacher, Skills, Faculty)

Б) Практическая часть.

1. Класс List (список). Может быть однонаправленный или двунаправленый, с фиктивной головой или нет.
   2 независимых класса – узел (Node) и список (могут быть друзьями).
   Член-данные для списка: указатель на голову, указатель на хвост (для двунаправленного)
   Методы: List (); List (int); List (int*, int) (из массива делаем список); List (const List &); ~List(); Node* FindKey(int) (на входе – ключ, на выходе – указатель на узел, если ключ в списке нашли, NULL – если ключа в списке нет); функции ввода/вывода (можно только перегрузить потоковые); сортировка; добавление элемента (в голову, хвост, на позицию, после ключа), удаление элемента (из головы, хвоста, с позиции, по ключу), нахождение max/min; проверка пустоты списка; очистка списка.
   Перегрузка операторов: = (присвоение); [ ] ; +(List) (в голову, в хвост); ==, != (сравнение списков); потокового ввода/вывода.
2. Классы Student и Teacher, наследники класса Person.
   а) Определить родительский класс Person (член-данные: string fio, address; int year; методы: конструкторы; void Input() (ввод данных с клавиатуры); void Show() (вывод данных на консоль).
   б) Член-данные класса Student: int group, kurs; string spec.
   Методы класса Student: конструкторы; void Input() (ввод данных с клавиатуры); void Show() (вывод данных на консоль), Get_group(); Get_kurs();
   Перегрузить потоковый вывод.
   в) Член-данные класса Teacher: string kafedra, prof.
   Методы класса Student: конструкторы; void Input() (ввод данных с клавиатуры); void Show() (вывод данных на консоль), Get_Kafedra(); Get_Prof();
   Перегрузить потоковый вывод.
3. Агрегированный класс Faculty.
   Член-данные класса Faculty: vector<Student> s; vector <Teacher> t; string long_name, short_name;
   Методы: конструкторы; добавление студента/преподавателя; удаление студента/преподавателя; вывод (всех, всех студентов, всех преподавателей, конкретной группы, конкретного курса, конкретной кафедры); получить все данные студента/преподавателя; вывод общего количества студентов/преподавателей.
4. БСШ, класс vector.
   Задать массив случайных целых чисел и
   а) выбрать из него в другой массив уникальные числа (т.е. встречающиеся только один раз); б) выбрать из него в другой массив числа, кратные 5.
   Задать массив строк и удалить из него самую длинную строку.
   Задать массив чисел Фибоначчи и выбрать из него в другой массив четные числа.
5. БСШ, класс list.
   Задать список фамилий и
   а) Добавить перед самой длинной фамилией фамилию fam; б) отсортировать его по возрастанию; в) найти и удалить самую длинную фамилию.
   Задать 2 упорядоченных списка фамилий, слить их в один общий упорядоченный список.
6. БСШ – класс set
   Задать 2 предложения. Выяснить, в каком предложении разных букв больше.

Для приобретения практического навыка объектно-ориентированного программирования, студентам на лабораторных занятиях предлагается реализовать следующие классы.

1) Класс «Комплексное число».
2) Класс «Массив».

3) Класс «Маршрут»
4) Класс «Строка»
5) Класс «Булев вектор произвольной длины».
6) Агрегированный класс «Булева матрица».

Критерии оценивания.

Коллоквиум проводится в письменном форме и оценивается «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Оценка «**отлично**» выставляется, если:

а) студент дал полный и развернутый ответ на теоретический вопрос;

б) код практического задания верен, оптимален (по скорости или по объему памяти), легко читаем, при написании кода использованы эффективные алгоритмы.

Оценка «**хорошо**» выставляется, если:

а) ответ студента на теоретический вопрос в целом полный, но имеются незначительные замечания;

б) код практического задания верен, но не оптимален (по скорости и по объему памяти), при написании кода использованы трудоемкие алгоритмы.

в) код практического задания верен, оптимален (по скорости или по объему памяти), легко читаем, при написании кода использованы эффективные алгоритмы, но студент выполнил не все практическое задание (но более 80%)

Оценка «**удовлетворительно**» выставляется, если:

а) ответ студента на теоретический вопрос не полный;

б) код практического задания содержит ошибки синтаксического характера.

в) студент выполнил менее 80% практического задания (но более 50%)

Оценка «**неудовлетворительно**» выставляется, если:

а) студент не ответил на теоретический вопрос или ответ студента на теоретический вопросы не полный и содержит серьезные ошибки;

б) практическое задание не выполнено или код практического задания содержит синтаксические и алгоритмические ошибки.

в) студент выполнил менее 50% практического задания.

Задания, выполняемые студентами во время лабораторных работ, оцениваются следующим образом:

- оценка «**отлично**» выставляется, если студент реализовал все методы класса; допускается реализация отдельных методов не самым оптимальным образом;
- оценка «**хорошо**» выставляется, если студент реализовал 90% методов или реализованы все методы, но для большей части методов использованы не самые эффективные алгоритмы;
- оценка «**удовлетворительно**» выставляется, если студент реализовал не менее 70% методов класса или реализованы все методы, но для большей части реализация построена на неэффективных алгоритмах, либо не учитывает появление исключений;
- оценка «**неудовлетворительно**» выставляется, если студент реализовал менее 70% методов класса, либо в реализациях не менее 20% методов имеются грубые ошибки алгоритмического характера.

Для каждого задания устанавливается срок выполнения. Задания, сданные позже установленного срока без уважительной причины, оцениваются на балл ниже реальной оценки.

Четвертый семестр

Элементы текущего контроля:
– посещаемость занятий;
–тесты;
– контрольные работы;
– практические задания;

Типовые задания для проведения текущего контроля успеваемости по дисциплине (ОПК-7, ИОПК-7.1, ИОПК-7.2, ИОПК-7.3, ИОПК-7.4)

Задание 1. Составить программу на C# для перевода чисел из одной системы счисления в другую. Системы счисления произвольные. Для этого сначала реализуем алгоритм перевода чисел из системы счисления N в десятичную систему счисления, потом из десятичной переводим в систему счисления M.

Если N = 10 или M = 10 это частный случай, его надо проверить.

Составляем подробный отчёт, в котором должен быть код с подробными комментариями и скрины результатов выполнения программы во всех нужных случаях, граничных и обычных. Отчёт прикрепляем в Moodle, лучше в формате pdf.

Задание 2. Реализовать алгоритм Ханойские башни для любого N.

Прикрепляем подробный отчёт, в котором должен быть код с подробными комментариями и скрины результатов выполнения программы, здесь тоже подробно и с комментариями

Задание 3. Дано арифметическое выражение в виде строки. Найти его значение.

Прикрепляем подробный отчёт, в котором должен быть код с подробными комментариями и скрины результатов выполнения программы во всех нужных случаях, показать, как вычисляются выражения с одинарным минусом, скобками, ошибками.

Задание 4. Создать небольшой телеграм-бот для указанной рассылки, используя язык программирования Python. Подобрать нужные библиотеки. Прикрепляем подробный отчёт, в котором должен быть код с подробными комментариями и ссылка на разработанный бот.

Задание 5. Междисциплинарная, задача из теории графов. Реализовать алгоритм раскраски графа для графов большой размерности. Подобрать нужные библиотеки. Прикрепляем подробный отчёт, в котором должно быть обоснование выбранных средств реализации, код с подробными комментариями и скрины результатов.

Критерии оценивания заданий 1-5

Критерии	0 баллов	1 балл	2 балла
Корректность постановки задачи (если есть)	Задача поставлена не корректно, непонятно что должна делать программа.	Задача сформулирована корректно, но с ошибками.	Задача сформулирована корректно.

Правильность выбора средств реализации	Средства реализации выбраны неправильно, задачу реализовать тяжело.	Средства реализации выбраны правильно, но существуют более подходящие средства.	Средства реализации выбраны правильно, задачу легко реализовать.
Правильность алгоритма	Алгоритм реализован неправильно, задача не решена	Алгоритм реализован так, что задача решена частично.	Алгоритм реализован правильно, задача решена
Правильность работы программы	Облачные технологии не использованы	Программа использует облачные технологии в недостаточном объёме.	Программа работает с облачными технологиями
Адекватность отчёта	Отчёт не сделан	Отчёт не даёт представления о решении задачи.	Отчёт даёт представление о решении задачи.

Кейс 1. Междисциплинарный, задача из алгебры. Решить задачу в небольших группах, затем обсудить представленные программы.

Найти НОД(f(x),g(x)) в конечном поле Z(p). Задачу можно усложнить, взяв поле $Z(p^k)$.

Решение кейса должно быть представлено группе и преподавателю в виде отчёта и доклада с презентацией, возможно взаимное оценивание работы групп.

Кейс 2. Междисциплинарный, статистика. Сбор статистических данных из сети Интернет и исследование их характеристик.

Выбрать тему для исследования, решить задачу и представить результат.

Пример темы: построить корреляцию между продолжительностью жизни и отношения к курению.

Решение кейса должно быть представлено группе и преподавателю в виде отчёта и доклада с презентацией, возможно взаимное оценивание работы групп.

Критерии оценивания кейсов 1, 2

Критерии	0 баллов	1 балл	2 балла
Корректность постановки задачи (если есть)	Задача поставлена не корректно, непонятно что должна делать программа.	Задача сформулирована корректно, но с ошибками.	Задача сформулирована корректно.
Правильность выбора средств реализации	Средства реализации выбраны неправильно, задачу реализовать тяжело.	Средства реализации выбраны правильно, но существуют более подходящие средства.	Средства реализации выбраны правильно, задачу легко реализовать.
Правильность алгоритма	Алгоритм реализован неправильно, задача не решена	Алгоритм реализован так, что задача решена частично.	Алгоритм реализован правильно, задача решена

Правильность работы программы	Облачные технологии не использованы	Программа использует облачные технологии в недостаточном объёме.	Программа работает с облачными технологиями
Адекватность отчёта	Отчёт не сделан	Отчёт не даёт представления о решении задачи.	Отчёт даёт представление о решении задачи.
Качество оформления презентации	Небрежное оформления презентации.	Оформление недостаточно продумано и аккуратно	Оформление продумано и аккуратно
Умение отвечать на вопросы	Студент не ответил на вопросы	Студент ответил только на часть вопросов	Студент ответил на все вопросы
Умение задавать вопросы	Студент не задаёт вопросы другим выступающим	Студент задал 2-3 вопроса другим выступающим	Студент активно задаёт вопросы другим выступающим

3. Оценочные материалы итогового контроля (промежуточной аттестации) и критерии оценивания (ИПК-7, ИОПК-7.1, ИОПК-7.2, ИОПК-7.3, ИОПК-7.4)

Третий семестр

Зачет в третьем семестре проводится в письменной форме по билетам. Билет состоит из двух частей. Первая часть представляет собой два теоретических вопроса. Ответы на вопросы даются в развернутой форме. Вторая часть состоит из практического задания. Ответы на вопросы второй части предполагают написание программного кода для поставленной задачи и анализ его работы. Оценивается оптимальность выбранного для решения задачи алгоритма и скорость его работы.

Продолжительность зачета 1,5 часа.

Результаты зачетной работы оцениваются «зачтено» или «не зачтено»

Оценка «**зачтено**» выставляется если

а) студент дал полный и развернутый ответ на теоретические вопросы или ответ студента на теоретические вопросы в целом полный, но имеются незначительные замечания;

б) код практического задания верен, оптимален (по скорости или по объему памяти), легко читаем, при написании кода использованы эффективные алгоритмы.

в) код практического задания верен, но не оптимален (по скорости и по объему памяти), при написании кода использованы трудоемкие алгоритмы.

г) код практического задания верен, оптимален (по скорости или по объему памяти), легко читаем, при написании кода использованы эффективные алгоритмы, но студент выполнил не все практическое задание (но более 80%)

Оценка «**не зачтено**» выставляется если

а) студент не ответил на теоретические вопросы (один или оба) или ответ студента на теоретические вопросы не полный и содержит серьезные ошибки;

б) практическое задание не выполнено или код практического задания содержит синтаксические и алгоритмические ошибки.

в) студент выполнил менее 50% практического задания

Студенты, сдавшие в течение семестра коллоквиумы на положительные оценки и выполнившие все лабораторные работы, получают зачет автоматически.

Четвертый семестр

Тест по основам C#

1. Объект является
    - экземпляром функции
    - экземпляром класса
    - экземпляром структуры
    - экземпляром делегата
    - экземпляром события
2. Типы по значению хранятся
    - В управляемой куче
    - В стеке
    - В неуправляемой куче
3. Ссылочные типы хранятся
    - В управляемой куче
    - В стеке
    - В неуправляемой куче
4. К типам по значению в C# относятся
    - классы
    - структуры
    - делегаты
    - перечисления
    - встроенные (базовые) типы
    - массивы
5. К ссылочным типам в C# относятся
    - классы
    - структуры
    - делегаты
    - перечисления
    - встроенные (базовые) типы
    - массивы
6. В C# к членам структуры по умолчанию устанавливается спецификация доступа
    - public
    - private
    - protected
    - internal
7. Статические методы класса могут обращаться
    - к не статическим методам класса
    - к статическим полям
    - к любым полям класса
    - к статическим методам класса
8. В C# к членам класса по умолчанию устанавливается спецификация доступа
    - private
    - public
    - protected
    - internal
9. В C# к членам интерфейса по умолчанию устанавливается спецификация доступа
    - private
    - public

protected
internal

10. Поля структуры можно инициализировать
    При объявлении
    В конструкторе с аргументами
    В конструкторе без аргументов
11. Допускается не инициализировать перед вызовом функции параметр с модификатором
    ref
    out
    static
12. Для передачи функции переменного количества параметров используется ключевое слово:
    out
    params
    refg
13. Для передачи функции параметров по ссылке необходимо использовать ключевое слово
    params
    out
    static
    ref
14. Перегруженные функции отличаются
    Типом возвращаемого значения.
    Списком принимаемых значений.
    И тем и другим.
    Именем
15. Вызов статической функции класса осуществляется с использованием
    Ссылки на объект
    Имени класса
    Имени структуры
16. Конструктор имеет тип возвращаемого значения
    void
    не имеет
    int
17. Конструктор по умолчанию может быть объявлен в
    Структуре
    Классе
    Интерфейсе
18. Конструктор с аргументами может быть объявлен
    Без спецификации доступа
    Со спецификацией public
    Со спецификацией private
    Со спецификацией protected
19. Конструкторов по умолчанию в структуре может быть объявлено
    один
    ни одного
    сколь угодно много
20. Константы инициализируются
    При объявлении
    С помощью конструктора по умолчанию
    С помощью конструктора с аргументами
21. Конструктор может вызвать конструктор

Производного класса
Базового класса
Своего класса

22. Для вызова конструктора базового класса необходимо использовать
имя базового класса
ключевое слово base
ключевое слово this

Тест оценивается следующим образом:
- Отлично – от 97% до 100% правильных ответов
- Хорошо – от 90% до 96% правильных ответов
- Удовлетворительно – от 70% до 89% правильных ответов
- Неудовлетворительно – менее 69% правильных ответов

Зачёт с оценкой выставляется по результатам проверки лабораторных работ и кейсов (70%), и оценки за тест (30%). При этом за задания и кейсы максимально можно набрать 14 баллов. Оценка выставляется следующим образом:
- Отлично – от 12 до 14 баллов
- Хорошо – от 9 до 11 баллов
- Удовлетворительно – от 6 до 8 баллов
- Неудовлетворительно – менее 5 баллов

4. Оценочные материалы для проверки остаточных знаний (сформированности компетенций)

Написать программу (ИПК-7, ИОПК-7.1, ИОПК-7.2, ИОПК-7.3, ИОПК-7.4)
1. Написать программу обработки числового массива (на языке C++, C#, Python).
2. Реализовать небольшой класс (на языке C++, C#).
3. Работа с файлами (на языке C++, C#, Python).

Информация о разработчиках

Самохина Светлана Ивановна, канд. физ.-мат. наук, доцент, доцент кафедры компьютерной безопасности института прикладной математики и компьютерных наук НИ ТГУ.