МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Новосибирский национальный исследовательский государственный университет»
(Новосибирский государственный университет, НГУ)

Физический факультет
Кафедра физико-технической информатики

УТВЕРЖДАЮ
Декан ФФ, д.ф.-м.н
В.Е.Блинов
«__» _____ 2022 г.

Рабочая программа дисциплины

ПРОГРАММИРОВАНИЕ НА C++ И PYTHON

направление подготовки: **03.03.02 Физика**
направленность (профиль): **Физическая информатика**

Форма обучения
Очная

Семестр	Общий объем	Виды учебных занятий (в часах)				Промежуточная аттестация (в часах)				
		Контактная работа обучающихся с преподавателем			Самостоятельная работа, не включая период сессии	Самостоятельная подготовка к промежуточной аттестации	Контактная работа обучающихся с преподавателем			
		Лекции	Практические занятия	Лабораторные занятия			Консультации	Зачет	Дифференцированный зачет	Экзамен
1	2	3	4	5	6	7	8	9	10	11
3	108	16	48		42				2	
Всего 108 часов / 3 зачётные единицы, из них: - контактная работа 66 часов										
Компетенции ОПК-3										

Ответственный за образовательную программу
д.ф.-м.н., проф. С. В. Цыбуля

Новосибирск, 2022

Содержание

Аннотация

к рабочей программе дисциплины
«Программирование на C++ и Python»
Направление: **03.03.02 Физика**
Направленность (профиль): Физическая информатика

Программа курса **«Программирование на C++ и Python»** составлена в соответствии с требованиями СУОС по направлению подготовки **03.03.02 Физика, направленность «Физическая информатика»,** а также задачами, стоящими перед Новосибирским государственным университетом по реализации Программы развития НГУ. Дисциплина реализуется на физическом факультете Федерального государственного автономного образовательного учреждения высшего профессионального образования Новосибирский национальный исследовательский государственный университет (НГУ) кафедрой физико-технической информатики. Дисциплина изучается студентами второго курса физического факультета в качестве одной из дисциплин по выбору вариативной части образовательной программы.

Цель курса – ознакомить студентов с современными подходами, применяемыми в программировании, и дать начальные навыки пользования языками программирования C++ и Python и средствами коллективной разработки программ.

Дисциплина нацелена на формирование у выпускника общепрофессиональных компетенций:

Результаты освоения образовательной программы (компетенции)	Индикаторы	Результаты обучения по дисциплине
ОПК-3. Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности.	**ОПК - 3.1.** Применяет различные источники информации для решения задач профессиональной сферы деятельности. **ОПК – 3.2.** Применяет основные приемы, возможности и правила работы со стандартными и специализированными программными продуктами при решении профессиональных задач. **ОПК – 3.3.** Применяет методологию поиска научной и технической информации в сети Интернет и специализированных базах данных.	**Знать** синтаксис и основные конструкции языков C++ и Python; основные части стандартной библиотеки C++: iostream, fstream, sstream, string, vector, set, map, algorithm, numeric, memory, iterator, functional; основные компоненты стандартной библиотеки языка Python; библиотеки для научных вычислений и визуализации данных: numpy, scipy, matplotlib, sympy; принципы написания программ в объектно-ориентированном подходе. **Уметь** работать с документацией языков программирования C++ и Python, с документацией библиотек этих языков; разрабатывать структуру программ на языках C++ и Python; писать и отлаживать программы на языках C++ и Python; пользоваться инструментами стандартных библиотек

Результаты освоения образовательной программы (компетенции)	Индикаторы	Результаты обучения по дисциплине
		языков C++ и Python; пользоваться библиотеками языка Python для анализа данных. **Владеть:** редактором VS Code, компилятором языка C++ из набора GCC, системой для сборки проектов Cmake, системой контроля версий git, интерпретатором python.

Курс рассчитан на один семестр. Преподавание дисциплины предусматривает следующие формы организации учебного процесса: лекции, практические занятия, самостоятельная работа студента и её контроль преподавателями с помощью заданий, дифференцированный зачёт.

Программой дисциплины предусмотрены следующие виды контроля:

Текущий контроль: задания для самостоятельного решения.

Промежуточная аттестация: дифференцированный зачёт.

Общая трудоемкость рабочей программы дисциплины составляет **108** академических часов / **3** зачетные единицы.

1. Перечень планируемых результатов обучения по дисциплине, соотнесённых с планируемыми результатами освоения образовательной программы.

Основной целью учебного курса «Программирования на C++ и Python» является ознакомить студентов с современными подходами, применяемыми в программировании, и дать начальные навыки пользования языками программирования C++ и Python и средствами коллективной разработки программ.

Курс помогает студентам преодолеть начальный порог вхождения в область современного программирования и получить первичные навыки использования языков C++ и Python. Выбор языков C++ и Python не случаен – это два самых популярных современных языка в области научных вычислений. Кроме того, использование C++ позволяет объяснить основные понятия объектно-ориентированного подхода на основе уже известного студентам языка C, который последовательно расширяется дополнительными возможностями, необходимыми для реализации концепций ООП. Мультиязычность курса помогает отделить концепции и понятия от их реализации в конкретном языке программирования. Знакомство с языком Python помогает студентам открыть для себя этот исключительно мощный, универсальный и простой в использовании инструмент для решения практически любых задач, где требуется написать программу. По итогам курса студенты должны быть способны к самостоятельному изучению языков и технологий в области программирования.

Результаты освоения образовательной программы (компетенции)	Индикаторы	Результаты обучения по дисциплине
ОПК-3. Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности.	**ОПК - 3.1.** Применяет различные источники информации для решения задач профессиональной сферы деятельности. **ОПК – 3.2.** Применяет основные приемы, возможности и правила работы со стандартными и специализированными программными продуктами при решении профессиональных задач. **ОПК – 3.3.** Применяет методологию поиска научной и технической информации в сети Интернет и специализированных базах данных.	**Знать** синтаксис и основные конструкции языков C++ и Python; основные части стандартной библиотеки C++: iostream, fstream, sstream, string, vector, set, map, algorithm, numeric, memory, iterator, functional; основные компоненты стандартной библиотеки языка Python; библиотеки для научных вычислений и визуализации данных: numpy, scipy, matplotlib, sympy; принципы написания программ в объектно-ориентированном подходе. **Уметь** работать с документацией языков программирования C++ и Python, с документацией библиотек этих языков; разрабатывать структуру программ на языках C++ и Python; писать и отлаживать программы на языках C++ и Python; пользоваться инструментами стандартных библиотек языков C++ и Python; пользоваться библиотеками языка Python для анализа данных.

Результаты освоения образовательной программы (компетенции)	Индикаторы	Результаты обучения по дисциплине
		Владеть: редактором VS Code, компилятором языка C++ из набора GCC, системой для сборки проектов Cmake, системой контроля версий git, интерпретатором python.

2. Место дисциплины в структуре образовательной программы.

Учебный курс «Программирования на C++ и Python» относится к вариативной части программы дисциплин бакалавриата.
Для успешного освоения курса «Программирования на C++ и Python» студенты должны обладать предварительными знаниями основ программирования.

3. Трудоёмкость дисциплины в зачётных единицах с указанием количества академических часов, выделенных на контактную работу обучающегося с преподавателем (по видам учебных занятий) и на самостоятельную работу.

Семестр	**Общий объем**	**Виды учебных занятий (в часах)**				**Промежуточная аттестация (в часах)**				
		Контактная работа обучающихся с преподавателем			Самостоятельная работа, не включая период сессии	Самостоятельная подготовка к промежуточной аттестации	Контактная работа обучающихся с преподавателем			
		Лекции	Практические занятия	Лабораторные занятия			Консультации	Зачет	Дифференцированный зачет	Экзамен
1	2	3	4	5	6	7	8	9	10	11
3	108	16	48		42				2	
Всего 108 часов / 3 зачётные единицы, из них: - контактная работа 66 часов										
Компетенции ОПК-3										

Реализация дисциплины предусматривает практическую подготовку при проведении следующих видов занятий, предусматривающих участие обучающихся в выполнении отдельных элементов работ, связанных с будущей профессиональной деятельностью: лекции, практические занятия, самостоятельная работа студента и её контроль преподавателями с помощью заданий, дифференцированный зачёт.

Программой дисциплины предусмотрены следующие виды контроля:

- текущий контроль успеваемости: задания для самостоятельного решения;
- промежуточная аттестация: дифференцированный зачёт.

Общая трудоемкость рабочей программы дисциплины составляет 3 зачетные единицы.

- занятия лекционного типа – 16 часов;
- практические занятия – 48 часов;
- самостоятельная работа обучающегося в течение семестра, не включая период сессии – 42 часа;
- промежуточная аттестация (дифференцированный зачёт) – 2 часа.

Объём контактной работы обучающегося с преподавателем (занятия лекционного типа, практические занятия, дифференцированный зачёт) составляет 66 часов.

4. Содержание дисциплины, структурированное по темам (разделам) с указанием отведённого на них количества академических часов и видов учебных занятий.

Общая трудоемкость дисциплины составляет 3 зачётные единицы, 108 академических часов.

№ п/п	Раздел дисциплины	Неделя семестра	Виды учебной работы, включая самостоятельную работу студентов и трудоёмкость (в часах)					Консультации перед экзаменом (часов)	Промежуточная аттестация (в часах)
			Всего	Аудиторные часы		Сам. работа во время занятий (не включая период сессии	Сам. работа во время промежуточной аттестации		
				Лекции	Практические занятия				
1	2	3	4	5	6	7	8	9	10
1.	Введение в ООП	1-4	25	4	6	9			
2.	Стандартная библиотека языка C++	5-6	12	2	12	4			
3.	Основы проектирования и совместной разработки	7-8	13	2	6	5			.
4.	Основы языка Python	9-10	14	2	6	6			
5.	Обзор стандартной библиотеки языка Python	11-12	14	2	6	6			
6.	Научные вычисления в Python	13-16	26	4	12	12			
7.	Дифференцированный зачёт		2						2
Всего			**108**	**16**	**48**	**42**			**2**

Программа и основное содержание лекций (16 часов)

Раздел 1. Введение в ООП (4 часа)

Классы и объекты в C++. От структур к классам. Инкапсуляция, наследование, полиморфизм. Конструкторы. Виртуальные функции. Более глубокое знакомство с классами в C++. Перегрузка функций и операторов. Ссылки. Способы передачи аргументов. Const. Работа с динамической памятью. Конструктор копирования и перемещения. Static.

Раздел 2. Стандартная библиотека языка C++ (2 часа)

Стандартная библиотека C++. Потоки, перегрузка операторов ввода-вывода. Исключения. Пространства имен. Шаблоны и обобщенное программирование. STL. Строки, контейнеры, итераторы, алгоритмы, функторы, умные указатели.

Раздел 3. Основы проектирования и совместной разработки (2 часа)

Основы проектирования и совместной разработки. ООП. Процесс разработки ПО. Каскадная, итеративная, гибкие модели. UML. Паттерны проектирования. Системы управления версиями. Системы с централизованным и распределенным репозиторием.

Раздел 4. Основы языка Python (2 часа)

Основы языка Python. Основы языка. Основные структуры данных. Объектно-ориентированное программирование на языке Python.

Раздел 5. Обзор стандартной библиотеки языка Python (2 часа)

Модули os, sys, string, decimal, fractions, collections, itertools, datetime.

Раздел 6. Научные вычисления в Python (4 часа)

Вычисления с numpy: массивы numpy, механизм broadcasting, линейная алгебра с numpyю. Визуализация данных с matplotlib: plot, errorbar, hist, scatter, countour, продвинутая настройка графиков. Модули библиотеки scipy: special, integrate, linalg, interpolate, optimize, stats, fftpack. Символьные вычисления с sympy

Программа практических занятий (48 часов)

Раздел 1. Введение в ООП (9 часов)

Занятие 1-3. Классы. Публичные и приватные поля и методы. Статические поля и методы. Конструктор, деструктор, специальные методы. Константные поля и методы. Перегрузка операторов. Ключевые слова delete и default. Наследование. Абстрактные типы. Виртуальные функции. Ключевые слова override и final. Динамическое выделение памяти. Использование new и delete. Умные указатели unique_ptr и shared_ptr. Идиома RAII. Обобщенное программирование. Шаблонные функции. Шаблонные классы.

Раздел 2. Стандартная библиотека языка C++ (12 часов)

Занятие 4-7. Работа с потоками ввода-вывода. Библиотеки iostream, fstream, sstream. Работа со строками в C++. Библиотека string. Контейнеры стандартной библиотеки C++: vector, list, array, deque, priority_queue, set, map, unordered_set, unordered_map. range-based for loop. Эффективная передача параметров в функцию: передача параметров в функцию по ссылке, константные параметры. Итераторы: типы итераторов, конструирование контейнеров с помощью итераторов. Алгоритмы стандартной библиотеки C++: for_each, sort, stable_sort, lower_bound, upper_bound, binary_search, copy_if, unique, any_of, all_of, none_of, copy_if, erase. Компараторы и предикаты. Лямбда-функции. Библиотека numeric: iota, accumulate, inner_product.

Раздел 3. Основы проектирования и совместной разработки (3 часа)

Занятие 8. Настройка рабочей среды: создание GitHub аккаунта, установка VS Code, mingw, git, Cmake. Компиляция программ с помощью gcc и Cmake. Освоение процедуры сдачи заданий. Выполнение задания "Hello, Classroom!".

Раздел 4. Основы языка Python (6 часов)

Занятие 9-10. Основы синтаксиса языка python. Базовые типы данных и встроенные контейнеры int, float, list, str, dict, set, tuple. Запись/чтение из файла. Подробнее о строках.: f-, r-, и b-строки, методы count, capitalize, isdigit, encode и др. Полезные модули стандартной библиотеки: библиотеки os и sys, аргументы командной строки и библиотека argparse, библиотеки datetime и calendar, библиотеки decimal и fraction

Раздел 5. Обзор стандартной библиотеки языка Python (6 часов)

Занятие 11-12. Объектно-ориентированная разработка с python. Продолжаем знакомство со стандартной библиотекой python. Основы работы с web: библиотеки wget, requests. Сериализация с библиотекой json. Временные файлы с библиотекой tempfile. Итераторы, генераторы. Ключевое слово yield.

Раздел 6. Научные вычисления в Python (12 часов)

Занятие 13-16. Вычисления с библиотекой numpy. Создание массивов numpy. Способы индексирования. Операции с массивами numpy, broadcasting. Случайные числа с numpy. Линейная алгебра с numpy. Визуализация данных с библиотекой matplotlib. Графики plot, errorbar. Гистограммы hist. Диаграммы рассеяния scatter. Контурные диаграммы contour. Настройка отображения графиков. Графика с библиотекой pygame. Структура программы с pygame, главный цикл. Отображение геометрических фигур. Обработка событий.

Самостоятельная работа студентов (42 часа)

Перечень занятий на СРС	Объем, час
Подготовка к практическим занятиям.	36
Изучение теоретического материала, не освещаемого на лекциях	6

5. Перечень учебной литературы.

5.1. Основная литература

1. Лутц, Марк. Изучаем Python : [эффективное объектно-ориентированное программирование : уровень подготовки читателей: средний : пер. с англ.] / Марк Лутц. 3-е изд. Москва ; Санкт-Петербург : Символ, 2009. 843 с. : ил., табл. ; 24 см.

5.2. Дополнительная литература

2. Саммерфилд, Марк. Программирование на Python 3 : подробное руководство : [пер. с англ.] / Марк Саммерфилд. Санкт-Петербург ; Москва : Символ, 2009. 607 с. : ил. ; 23 см. (High tech) .

6. Перечень учебно-методических материалов по самостоятельной работе обучающихся.

Самостоятельная работа студентов поддерживается следующими учебными пособиями:

3. Структурный и объектно-базированный подходы в программировании на примере языков С и С++ : К.Ф. Лысаков, А.А. Дунаев ; Федер. агентство по образованию, Новосиб. гос. ун-т, Физ. Фак. Новосибирск : Редакционно-издательский центр НГУ, 2007. ISBN 978-5-94356-584-7.

7. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины.

Для освоения дисциплины используются следующие ресурсы:
- электронная информационно-образовательная среда НГУ (ЭИОС);
- образовательные интернет-порталы;
- информационно-телекоммуникационная сеть Интернет.

7.1 Современные профессиональные базы данных

Не используются.

7.2. Информационные справочные системы

Не используются.

8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине.

Для обеспечения реализации дисциплины используется стандартный комплект программного обеспечения (ПО), включающий регулярно обновляемое лицензионное ПО Windows и MS Office.

Также используется среда программирования Atmel Studio.

9. Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине.

Для реализации дисциплины используются специальные помещения:

1. Учебные аудитории для проведения занятий лекционного типа, практических занятий, курсового проектирования (выполнения курсовых работ), групповых и индивидуальных консультаций, текущего контроля, промежуточной и итоговой аттестации.

2. Помещения для самостоятельной работы обучающихся.

Учебные аудитории укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой с возможностью подключения к сети "Интернет" и обеспечением доступа в электронную информационно-образовательную среду НГУ.

Материально-техническое обеспечение образовательного процесса по дисциплине для обучающихся из числа лиц с ограниченными возможностями здоровья осуществляется согласно «Порядку организации и осуществления образовательной деятельности по образовательным программам для инвалидов и лиц с ограниченными возможностями здоровья в Новосибирском государственном университете».

10. Оценочные средства для проведения текущего контроля и промежуточной аттестации по дисциплине.

10.1 Порядок проведения текущего контроля и промежуточной аттестации по дисциплине

Текущий контроль

Текущий контроль успеваемости осуществляется в ходе сдачи заданий для самостоятельного решения по каждой теме, изучаемой на практике. За каждое задание обучающийся получает оценку.

Промежуточная аттестация

Освоение компетенций оценивается согласно шкале оценки уровня сформированности компетенции. Положительная оценка по дисциплине выставляется в том случае, если заявленная компетенция ОПК-3 сформирована не ниже порогового уровня в части, относящейся к формированию способности использовать специализированные знания в области программирования на C++ и Phyton в профессиональной деятельности.

Окончательная оценка работы студента в течение семестра происходит на дифференцированном зачёте в конце семестра. Он выставляется по результатам сдачи заданий для самостоятельного решения. Итоговая оценка рассчиитывается как средняя арифметическая из оценок, полученных за сдачу каждого из заданий.

Вывод об уровне сформированности компетенций принимается преподавателем. Положительная оценка ставится, когда все компетенции освоены не ниже порогового уровня. Оценки «отлично», «хорошо», «удовлетворительно» означают успешное прохождение промежуточной аттестации.

Соответствие индикаторов и результатов освоения дисциплины

Таблица 10.1

Индикатор	Результат обучения по дисциплине	Оценочные средства

ОПК - 3.1. Применяет различные источники информации для решения задач профессиональной сферы деятельности.	**Знать** синтаксис и основные конструкции языков C++ и Python; основные части стандартной библиотеки C++: iostream, fstream, sstream, string, vector, set, map, algorithm, numeric, memory, iterator, functional; основные компоненты стандартной библиотеки языка Python; библиотеки для научных вычислений и визуализации данных: numpy, scipy, matplotlib, sympy; принципы написания программ в объектно-ориентированном подходе.	Решение задач. Дифференцированный зачет.
ОПК – 3.2. Применяет основные приемы, возможности и правила работы со стандартными и специализированными программными продуктами при решении профессиональных задач.	**Уметь** работать с документацией языков программирования C++ и Python, с документацией библиотек этих языков; разрабатывать структуру программ на языках C++ и Python; писать и отлаживать программы на языках C++ и Python; пользоваться инструментами стандартных библиотек языков C++ и Python; пользоваться библиотеками языка Python для анализа данных.	Решение задач. Дифференцированный зачет.
ОПК – 3.3. Применяет методологию поиска научной и технической информации в сети Интернет и специализированных базах данных.	**Владеть:** редактором VS Code, компилятором языка C++ из набора GCC, системой для сборки проектов Cmake, системой контроля версий git, интерпретатором python.	Решение задач. Дифференцированный зачет.

10.2 Описание критериев и шкал оценивания индикаторов достижения результатов обучения по дисциплине «Программирование на C++ и Python».

Таблица 10.2

Критерии оценивания результатов обучения	Планируемые результаты обучения (показатели достижения заданного уровня освоения компетенций)	Уровень освоения компетенции			
		Не сформирован (0 баллов)	Пороговый уровень (3 балла)	Базовый уровень (4 балла)	Продвинутый уровень (5 баллов)
1	2	3	4	5	6

Полнота знаний	ОПК-3.1	Уровень знаний ниже минимальных требований. Имеют место грубые ошибки.	Минимально допустимый уровень знаний. Допускается значительное количество негрубых ошибок.	Уровень знаний соответствует программе подготовки по темам/разделам дисциплины. Допускается несколько негрубых/ несущественных ошибок. Не отвечает на дополнительные вопросы.	Уровень знаний соответствует программе подготовки по темам/разделам дисциплины. Свободно и аргументированно отвечает на дополнительные вопросы.
Наличие умений	ОПК-3.2	Отсутствие минимальных умений. Не умеет решать стандартные задачи. Имеют место грубые ошибки.	Продемонстрированы частично основные умения. Решены типовые задачи. Допущены негрубые ошибки.	Продемонстрированы все основные умения. Решены все основные задания с негрубыми ошибками или с недочетами.	Продемонстрированы все основные умения. Решены все основные задания в полном объеме без недочетов и ошибок.
Наличие навыков (владение опытом)	ОПК-3.3	Отсутствие владения материалом по темам/разделам дисциплины. Нет навыков в решении стандартных задач. Наличие грубых ошибок.	Имеется минимальный набор навыков при решении стандартных задач с некоторыми недочетами.	Имеется базовый набор навыков при решении стандартных задач с некоторыми недочетами.	Имеется базовый набор навыков при решении стандартных задач без ошибок и недочетов. Продемонстрированы знания по решению нестандартных задач.

10.3 Типовые контрольные задания и материалы, необходимые для оценки результатов обучения

Список заданий для самостоятельного решения

Задача 1. Класс LorentzVector

Напишите класс LorentzVector, реализующий Лоренц-векторы и содержащий

1. Конструктор по умолчанию
2. Конструктор, принимающий четыре числа double
3. Методы чтения компонент вектора
4. Методы модификации компонент вектора
5. Метод вывода компонент вектора в стандартный поток
6. Методы, реализующие сложение, вычитание, скалярное произведение Лоренц-векторов
7. Метод, реализующий преобразование Лоренца в систему координат, движущуюся со скоростью вдоль оси.

Указания:

- Определение классов стандартного вывода находятся в заголовочном файле <iostream>
- Реализуйте инкапсуляцию данных (работа со значениями компонент вектора производится только с помощью методов класса)
- Методы, не изменяющие компоненты вектора, объявляйте константными

• Избегайте копирования при передаче вектора в качестве аргумента метода (в методах сложения, вычитания и скалярного произведения). Используйте конструкцию

• В проекте должно быть три файла:

◦ LorentzVector.h – объявление класса

◦ LorentzVector.cpp – реализация методов класса

◦ main.cpp – содержит функцию main, в которой используется весь функционал класса LorentzVector

Задача 2. Перегрузка операторов для LorentzVector

Для класса LorentzVector перегрузите следующие операторы:

1. Сложение ()
2. Вычитание ()
3. Унарный минус ()
4. Умножение (), выполняющий скалярное произведение
5. Оператор вывода в стандартный поток

Указания:

• Создайте для этой задачи новый проект

• Покажите работу всех операторов в вашей функции main

• Оператор вывода в стандартный поток не следует делать методом класса LorentzVector. Остальные операторы можно реализовать в виде методов класса LorentzVector.

Задача 3. STL vector

В этом задании Вам предстоит поработать с шаблонным классом библиотеки STL std::vector. Задача состоит из следующих этапов:

1. Создать объект std::vector<int>
2. Заполнить этот объект случайными числами от до
3. Найдите количество элементов вектора, со значением больше некоторого порога A
4. Выполните сортировку массива
5. Выведите отсортированный массив в стандартный поток вывода
6. (Дополнительное задание). Повторите это упражнение для вектора трехмерных векторов. В качестве параметра сортировки и сравнения используйте длину вектора.

Подумать: как выполнить сортировку массива в порядке убывания?

Указания:

• В стандарте C++11 случайные числа можно генерировать следующим образом (заголовочный файл <random>):

• можно инициализировать с помощью метода seed(). Случайную инициализацию можно выполнить с помощью библиотеки <chrono>:

• Для заполнения вектора используйте алгоритм std::generate из библиотеки <algorithm>

• Используйте алгоритм std::count_if из библиотеки <algorithm>

• Чтобы применить алгоритм std::sort для сортировки вектора трехмерных векторов, достаточно перегрузить оператор <

• Реализуйте считывание параметров из файла или из стандартного потока

Задача 4. Скобки

Последовательность скобок может быть сбалансирована или нет. Например, [{([][])}[]] – сбалансированная последовательность, а {[]) – нет. Прочитайте строку из файла и определите сбалансированы ли скобки в этой строке. Учитывайте три вида скобок: {}, [] и ().

Указания:

• Для проверки баланса используйте класс std::stack<char>

• Обработайте невозможность чтения файла через исключение (блок try catch)

Input: Строка длиной от до символов. Не все символы могут быть скобками

Output: Сообщение “Balanced”, если скобки сбалансированы и “Not balanced” в противном случае

Задача 5. Иерархия классов

Разработайте иерархию классов с базовым классом и как минимум двумя наследниками. В базовом классе должен быть абстрактный виртуальный метод, реализованный в наследниках. Создайте гетерогенный список, содержащий объекты классов-наследников. Выведите объекты в стандартный вывод. Реализуйте сортировку списка по различным параметрам по возрастанию и по убыванию.

Пример иерархии: геометрические фигуры. Базовый интерфейс: , ,

Указание: алгоритм std::sort может использовать произвольную функцию сравнения для сортировки. Смотрите документацию (www.cplusplus.com/reference/algorithm/sort).

Часть 2. Python

Задание 1. Простые числа

Найдите и запишите в текстовый файл все простые числа до

Задание 2. Решить две задачи из следующего списка

2.1. Работа с массивом

a. Найти среднее, максимальное, минимальное значения, среднеквадратичное отклонение, пятый центральный момент

b. Отсортировать массив действительных чисел по возрастанию

c. Отсортировать массив комплексных чисел a) по модулю b) по убыванию действительной части

d. Отсортировать список строк по длине

e. Отсортировать кортеж строк в лексикографическом порядке

2.2. Вычислить число методом Монте-Карло. Сгенерировать случайные точки в квадрате. Подсчитать долю попавших в круг с радиусом и оценить значение. Какова точность такой

оценки? Сколько точек необходимо сгенерировать, чтобы достоверно получить пять знаков числа после запятой?

2.3. Реализовать клеточный автомат «Игра Жизнь»

2.4. Найти наибольшую общую подпоследовательность двух списков. Максимальная длина списка. Пример: наибольшая подпоследовательность последовательностей и является последовательность. Реализовать алгоритм динамического программирования (смотрите, например, статью Wikipedia «Наибольшая общая подпоследовательность»). Проверить правильность работы алгоритма прямым перебором.

Задание 3. Однострочники. Решить пять задач из следующего списка. Решение каждой задачи должно состоять из одной строки длиной не больше 80 символов (import не входит в ограничение)

3.1. Сложить все числа от одного до

3.2. Вычислить факториал

3.3. Удвоить все нечетные числа в списке

3.4. Найти элементы списка, отсутствующие в другом

3.5. Вывести названия всех клеточек поля для игры в морской бой

3.6. Найти количество четных чисел от 1 до N начинаются на 793 и не кратны 7

3.7. Считать строчку с клавиатуры и развернуть

3.8. Подсчитать количество букв “z” в файле

3.9. Вычислить константу с точностью 20 знаков

Задание 4. Фильтрация сигнала

Используя пакет, выполните следующие действия:

- Смоделировать сигнал из трёх некратных частот, добавить случайный шум
- Вычислить спектр сигнала, построить график, оси в единицах Гц
- Избавиться от шума в исходном сигнале посредством удаления «лишних» частот в Фурье-спектре
- Построить график исходного, зашумленного и фильтрованного сигналов с легендой

Часть 3. Проекты

В этой части приведены описания нескольких возможных проектов. Вы можете сами придумать тему проекта и, после обсуждения с преподавателем и его одобрения вашего предложения, заниматься проектом не из этого списка.

Проект. Двумерный идеальной газ

Численно промоделировать движение круглых частиц радиуса. Частицы находятся:

a) В прямоугольном ящике размером

b) В круглом ящике радиуса

Частицы упруго рассеиваются при столкновениях и упруго отталкиваются от стенок ящика. Начальные скорости равномерно распределены по углу и постоянны по модулю.

- Нарисовать положения всех частиц через время
- Анимировать движение частиц
- Дорисовать частицам стрелки – векторы скоростей
- Реализовать вытекание частиц через небольшое отверстие в ящике и нарисовать график зависимости количества частиц внутри ящика от времени

Проект. Просачивание (percolation)

Каждая клетка двумерной прямоугольной сетки размера может находиться в открытом или закрытом состоянии. Назовём «условием просачивания» состояние, при котором верхний и нижний края сетки могут быть соединены непрерывным путём через открытые клетки (смотрите рисунок).

Ваша задача состоит в моделировании такой системы. Необходим быстрый алгоритм для определения условия просачивания. Замечательным свойством такой системы является существование «фазового перехода»: существует пороговое значение вероятности открытости клетки. При условии 1 просачивание происходит почти наверняка, а при условии 2 просачивание почти наверняка не происходит. Аналитическое изучение системы предсказывает существование порога, но не даёт его значение.

Исследуйте систему и определите величину порога.

Для детектирования просачивания создайте структуру данных union-find (она же disjoint-set, или «система непересекающихся множеств»). Эта структура описана, например, в Wikipedia.

При работе в паре один человек может заниматься алгоритмами и логикой работы, а второй работать над визуализацией сетки и работы алгоритма поиска просачивания.

Проект. Игра «Пятнашки»

В этом проекте предлагается написать решатель известной головоломки «Пятнашки». Если Вы не помните правил этой головоломки, можете их найти, например, в Wikipedia. Результатом работы Вашего алгоритма должна быть последовательность оптимальных шагов, приводящих к собранной позиции, либо вывод сообщения о том, что позиция неразрешима.

Интересный факт состоит в том, что ровно половина позиций являются неразрешимыми. Чтобы получить из любой разрешимой позиции неразрешимую, достаточно поменять местами две соседние костяшки.

Для поиска оптимального решения предлагается использовать алгоритм. Этот алгоритм предлагает выбирать ход, который максимально приближает позицию к собранной. Одна из возможных количественных характеристик близости к собранной позиции – сумма Манхэттенских расстояний всех костяшек между их текущим положением и положением в собранной позиции. (Манхэттенским расстоянием называется сумма горизонтальных и вертикальных расстояний, подобно расстоянию между двумя пунктами в городе с сетью перпендикулярных дорог).

Более подробное описание алгоритма:

- Дана текущая позиция (на первом шаге исходная позиция)
- Проверяем не является ли позиция собранной (если является, то печатаем решение)
- Для текущей позиции находим все позиции, в которые можно перейти за один ход, исключая позицию, из которой получена текущая
- Выбираем из этих позиций позицию с наименьшей дистанцией до собранной позиции (при

нескольких вариантах выбираем любой). Для этого удобно использовать структуру данных.

- Повторяем действия в цикле

Как быть с неразрешимыми позициями?

Перед написанием кода хорошо подумайте над необходимыми классами и их интерфейсами. Обсудите это с преподавателем.

При выполнении проекта в паре, один студент может заниматься описанием доски (включая визуализацию), а второй – алгоритмом решения.

Оценочные материалы по промежуточной аттестации, предназначенные для проверки соответствия уровня подготовки по дисциплине требованиям СУОС, хранятся на кафедре-разработчике РПД в печатном и электронном виде.

Лист актуализации рабочей программы
по дисциплине «Программирование на C++ и Python»
по направлению подготовки 03.03.02 Физика
Профиль «Общая и фундаментальная физика»

№	Характеристика внесенных изменений (с указанием пунктов документа)	Дата и № протокола Учёного совета ФФ НГУ	Подпись ответственного