Министерство науки и высшего образования Российской Федерации Министерство науки, высшего образования и инноваций Кыргызской Республики

Государственное образовательное учреждение высшего профессионального образования Кыргызско-Российский Славянский университет имени первого Президента Российской Федерации Б.Н. Ельцина

Естественно-технический факультет

Кафедра Информационных и вычислительных технологий

Фонд оценочных средств

по дисциплине «Спецкурс по современным информационным технологиям»

Уровень высшего образования

БАКАЛАВРИАТ

Направление подготовки *11.03.04 - РФ, 691000 – КР*

Электроника и наноэлектроника

(код и наименование направления подготовки)

Квалификация

бакалавр

Фонд оценочных средств предназначен для контроля знаний, обучающихся по направлению подготовки 11.03.04 - РФ, 691000 – КР Электроника и наноэлектроника по дисциплине «Спецкурс по современным информационным технологиям».

Фонд оценочных средств рассмотрен и утвержден на заседании кафедры Информационных и вычислительных технологий

Заведующий кафедрой
д.т.н., проф. Лыченко Н.М.

Исполнители (разработчики):

Д.т.н., проф. каф. ИВТ Лыченко Н.М.

К.т.н., доцент каф. ИВТ Верзунов С.Н.

СОГЛАСОВАНО:
и.о. декана

Комарцов Н.М.

Приложение 1. Рекомендации для выполнения самостоятельной работы

Проект должен соответствовать следующим критериям:

• Он должен быть сопоставим по размеру и сложности с любым другим научным проектами, которые вы сделали в этом курсе (практические работы со 2 по 5).

• Это должно повлечь за собой написание как минимум 200 строк исходного компьютерного кода. Это может быть одна новая программа, или несколько более коротких программ, или значительное улучшение одной или нескольких программ, которые вы уже написали.

• Следует изучить научную сторону вопроса. Это может быть еще одна тема в кинематике, электронике, или тема из какой-то другой отрасли физики или тема из другой области естественных наук (биология, геоэкология и т. д.).

• Основная цель(и) проекта должна быть чем-то, что вы не могли бы осуществить без компьютера. То есть проект должен использовать вычисления нетривиальным образом.

• Вы должны приложитьзначительные усилия для анализа и интерпретации, а не просто написать компьютерный код. (Великолепный кусок кода сам по себе не является приемлемым проектом.)

• «Результатами» будут одна или несколько оригинальных компьютерных программ, отчет, полностью описывающий проект, и очень краткая презентация для защиты

• Вы должны выполнять свой окончательный проект индивидуально, а не с партнером из группы.

Тема проекта должна быть одобрена преподавателем.

Вы можете придумать отличную тему проекта, просто подумав творчески. Кроме того, не стесняйтесь обращаться к учебникам (например, тем, которые перечислены в РПД и другим источникам идей.

Хотя вы должны использовать язык Python для своего кодирования, если это возможно, вы может использовать другие вычислительные языки или среды, если у вас есть на то веские основания, при условии, что вы получите разрешение преподавателя заранее.

Пожалуйста, подготовьте 5-10-минутную презентацию (в зависимости от доступного времени) и опишите основные моменты вашего проекта. Это требуется для защиты проекта на практическом занятии в конце семестра. Планируйте использовать проектор, либо запустив моделирование в режиме реального времени, либо показав несколько слайдов PowerPoint, или же и то и другое.

Из-за нехватки времени ваша презентация не сможет покрыть все, что вы сделали. Так что подумайте, какие части вашей работы будут наиболее интересны аудитории, и сосредоточьтесь на них.

Ваш отчет должен быть более полным описанием вашего проекта.

Нужно разделить отчет на три основные части:

1. Введение, в котором вы объясняете цели и мотивы вашего проекта. На какие вопросы вы пытались ответить или какие проблемы вы пытались решить? Чем интересен этот проект?

2. Описание того, как вы выполняли проект. С чего вы начали? Какие были главные препятствия, и как вы их преодолели? Это будет вероятно самой длинной частью вашего отчета, но постарайтесь удержать интерес читателя сосредоточив внимание на наиболее важных вопросах, которые возникли, а не методически описывая все, шаг за шагом. Вы можете или не можете найти это целесообразным включить некоторые короткие фрагменты кода, но старайтесь избегать длинных отрывков кода. 75

3. Ваши результаты. Попробуйте представить их как можно более наглядно, с большим количеством иллюстраций. Короткие таблицы числовых результатов также иногда уместны.

Помимо этих трех основных разделов, необходимо в конце добавить краткие выводы с итогами проделанной работы.

Приложение 2. Типовые задания для проведения практических работ

Практическое задание №1

Первая цель в изучении любой новой среды компьютерного программирования всегда одна и та же: напишите и запустите программу для печати (или отображения) краткого сообщения, традиционно «Привет, мир!». Зачем это нужно? Потому что шаги, необходимые для написания и запуска даже самой простой программы, могут быть довольно сложными. В некоторых средах программирования вам может потребоваться установить программное обеспечение, настроить программное обеспечение для работы с системой каталогов вашего компьютера, а затем научиться использовать различные программные средства для редактирования, компиляции, компоновки и запуска вашей программы. Затем вам нужно иметь достаточно знаний о языке программирования и о связанных программных библиотеках для создания желаемого типа вывода, чтобы ввести код, необходимый для получения этого вывода. Количество вещей, которые могут пойти не так в течение всего этого процесса, огромно. К счастью, система GlowScript-VPython является одной из самых простых из всех возможных сред программирования для начала работы. 1. Запустите веб-браузер (рекомендуется Google Chrome, но большинство других должно работать) и перейдите на сайт glowscript.org.

2. Войдите, используя учетную запись Google. Ваши учетные данные электронной почты WSU должны работать, или вы можете использовать личный аккаунт Google.

3. Перейдите по ссылке, где написано «ваши программы здесь». 4. Нажмите на ссылку «Создать новую программу». В появившемся диалоговом окне введите заголовок «MakingShapes», затем нажмите кнопку «Создать». Теперь вы увидите пустое пространство для редактирования, за исключением строки «GlowScript2.7 VPython» (возможно, с другим номером версии). Нажмите на пустое пространство под этой строкой и введите следующее: дословно: print («Здравствуйте, GlowScript-VPython!») (На самом деле вы можете поместить практически любое понравившееся сообщение в кавычки.) Затем нажмите ссылку «Запустить эту программу» в верхней части страницы. Ваше пространство редактирования должно затем исчезнуть, заменившись новой текстовой областью со словами «Hello, GlowScript-VPython!» (Или любым другим сообщением, которое вы добавили в свой код).

(Наиболее вероятным камнем преткновения является вход в учетную запись Google. Если все прошло гладко, но ваша программа не работает, обязательно дважды проверьте орфографию, прописные буквы и знаки препинания. GlowScript-VPython - это облачная система который хранит ваши программы на серверах Google. Интерфейс прост, и я надеюсь, что у вас не возникнет проблем с тем, как создавать новые программы, копировать, переименовывать и удалять их, а также организовывать их в папки. Если у вас есть какие-либо вопросы по поводу этих процедур, обязательно спросите их у инструктора. Теперь давайте рассмотрим вашу двухстрочную программу немного подробнее. Нажмите на ссылку «Изменить эту программу», чтобы снова увидеть ваш код. Первая строка, которую вам даже не нужно набирать, сообщает системе GlowScript, какой язык (VPython)

и версию (2.7) вы хотите использовать. Эта строка обязательна, и для этого курса вы должны всегда оставлять ее в покое. Мы говорим, что следующая набранная вами строка «вызывает функцию печати и передает ей строку текстовых символов». Извините за жаргонные слова call, function, pass и строка. Если вы опытный программист, то эти слова, вероятно, уже есть в вашем словаре; если вы пишете код впервые, вам нужно обратить внимание на эти слова и практиковаться в их правильном использовании. А пока обратите внимание на два аспекта синтаксиса Python:
• Текстовая строка заключена в двойные кавычки. Одинарные кавычки также сработали бы, если совпадают начальная и конечная кавычки.
• Информация, передаваемая в функцию (ее параметр), заключена в скобки. В этом случае есть единственный параметр (строка), но мы скоро увидим примеры функций, которые принимают несколько параметров, разделенных запятыми. Идея функции состоит в том, чтобы скрыть детали того, что происходит, из вашего кода (или, по крайней мере, из этой части), так что все, что вам нужно знать, это имя функции, какой параметр (ы) для передачи и что функция делает, но не как она работает. Функция print должна сделать очень много, чтобы эти слова появились на вашем экране, но вам не нужно знать подробности.

Практическое задание №2

Идея моделирования состоит в том, чтобы запрограммировать законы физики в компьютер, а затем позволить компьютеру рассчитать, что происходит во времени, шаг за шагом в будущем. В этом курсе эти законы обычно будут законами движения Ньютона, и наша цель будет заключаться в том, чтобы предсказать движение одного или нескольких объектов, подверженных различным силам. Моделирование позволяет нам делать это для любых сил и любых начальных условий, даже когда не существует явной формулы для движения. В этом проекте вы будете моделировать движение снаряда, сначала в одном измерении, а затем в двух измерениях. Когда движение является чисто вертикальным, состояние снаряда определяется его положением *y* и его скоростью v_y. Эти величины связаны с помощью формулы:

$$v_y = \frac{dy}{dt} \approx \frac{\Delta y}{\Delta t} = \frac{y_{\text{final}} - y_{\text{initial}}}{\Delta t}. \quad (2.1)$$

В компьютерном моделировании мы уже знаем текущее значение *y* и хотим предсказать будущее значение. Итак, давайте решим это уравнение для

$$v_y = \frac{dy}{dt} \approx \frac{\Delta y}{\Delta t} = \frac{y_{\text{final}} - y_{\text{initial}}}{\Delta t}. \quad (2.2)$$

Аналогично, мы можем предсказать будущее значение v_y, если мы знаем текущее значение, а также ускорение:

$$v_{y,\text{final}} \approx v_{y,\text{initial}} + a_y \, \Delta t.$$ (2.3)

Эти уравнения справедливы для любого движущегося объекта. Для снаряда, движущегося вблизи земной поверхности без сопротивления воздуха, $a_y = -g$ (принимая направление + y вверх). В общем, ay задается вторым законом Ньютона,

$$a_y = \frac{\sum F_y}{m},$$ (2.4),

где *m* - масса объекта, а различные силы могут зависеть от y, v_y или от обоих. В компьютерном моделировании одномерного движения Идея состоит в том, чтобы начать с состояния частицы при $t = 0$, затем использовать уравнения с 2.2 по 2.4 для вычисления и v_y при $t = \Delta t$, затем повторить расчет для следующего интервала времени и так далее.

Практическое задание №3

В этой работе вы изучите поведение маятника. Нет лучшего примера системы, которая поначалу кажется простой, но оказывается, что она содержит разные уровни сложности. На рисунке 3.1 показана базовая установка: неподвижный шарнир, безмассовый стержень длиной *L* и точечная масса m на конце, качающиеся в плоскости страницы. Проще всего проанализировать движение с точки зрения крутящего момента и углового ускорения; Напомним, что угловая версия второго закона Ньютона

$$\sum \tau = I\alpha.$$ (3.1)

В левой части этого уравнения находится сумма всех моментов, действующих на объект. С правой стороны, *I* - инерция вращения объекта, просто mL^2 для нашего маятника. Угловое ускорение α определяется аналогично обычному ускорению:

$$\alpha = \frac{d\omega}{dt} = \frac{d^2\theta}{dt^2},$$ (3.2)

где ω - угловая скорость, а ϑ предполагается в радианах.

Из диаграммы видно, что момент силы тяжести равен

$$\tau_g = -L|\vec{F}_g|\sin\theta = -Lmg\sin\theta,$$ (3.3)

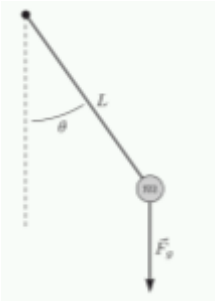


Рисунок 3.1: Простой маятник массы *m* и длины *L*, на который действует гравитационный сила F_g.

где знак минус указывает, что крутящий момент отрицателен (по часовой стрелке), когда θ положительный, и наоборот. Если нет трения или другого вращающего момента, то закон Ньютона (уравнение 3.1) гласит просто:

$$-Lmg\sin\theta = mL^2\alpha, \qquad \text{or} \qquad \alpha = -\frac{g}{L}\sin\theta. \quad (3.4)$$

Хотя приведен рисунок для небольшого положительного значения θ, это уравнение справедливо для всех углов – даже углов больше 90◦, если стержень жесткий.

Практическое задание №4

В этой работе будет моделироваться движение планет, вращающихся вокруг Солнца. Единственной важной силой в этой ситуации является гравитация. Согласно Ньютону, гравитационная сила между двумя объектами всегда положительна:

$$|\vec{F}_g| = \frac{Gm_1m_2}{r_{12}^2}, \quad (4.1)$$

где m_1 и m_2 – массы двух объектов, а r_{12} – расстояние между ними. Связанная с этим потенциальная энергия:

$$U_g = -\frac{Gm_1m_2}{r_{12}}. \quad (4.2)$$

К этой формуле нужно привыкнуть, потому что она всегда отрицательна. Однако важно то, что по мере того, как два объекта расходятся друг от друга, их потенциальная энергия увеличивается (становится менее отрицательной).

Начнем с одной планеты, вращающейся вокруг Солнца. Потому что Солнце более массивно, по сравнению с другой планетой. В этом случае можно поставить Солнце в начало системы координат, и силы на планете упростится.

Используйте уравнение 4.1 и соответствующую диаграмму, чтобы найти компоненты *x* и *y* гравитационной силы Солнца на планете в планетоцентрической системе координат. Убедитесь, что у вас есть правильные формулы, прежде чем пытаться написать любой код.

Практическое задание №5

Если возможно моделировать три взаимно взаимодействующих объекта, почему бы не смоделировать больше чем три? Как вы скоро увидите, нет особых трудностей при моделировании произвольного числа взаимодействующих частиц. Кроме того, современные компьютеры достаточно быстры, чтобы имитировать движение сотен частиц и анимировать это движение с разумной частотой кадров во время расчетов.

Основная трудность при разработке моделирования множества частиц заключается не в самой симуляции, а скорее в том, чтобы решить, что мы хотим извлечь из нее. Прогнозирование отдельные траектории сотен частиц, как правило, нецелесообразно, потому что траектории хаотичны. Даже если бы мы могли сделать такие прогнозы, огромное количество данных будет очень сложно для понимания. Вместо этого мы сосредоточимся на более высоком уровне абстракции. Основной целью будет узнать, как знакомые свойства вещества возникают из движений на молекулярном уровне.

Даже для самых простых молекул не существует простой точной формулы для межмолекулярной силы. К счастью, нам не нужна точная формула:



Рис. 5.1: Когда две молекулы находятся рядом друг с другом, существует слабая сила притяжения между ними. Когда они подходят слишком близко, сила становится отталкивающей.

Примерная формула с правильным общим поведением даст нам полезные результаты. Формула, которая чаще всего используется для моделирования взаимодействия между атомами газа – это потенциал Леннарда-Джонса:

$$U(r) = 4\epsilon\left[\left(\frac{r_0}{r}\right)^{12} - \left(\frac{r_0}{r}\right)^{6}\right]. \quad (5.1)$$

Это формула для потенциальной энергии, *U (r)*, в терминах расстояния *r* между центрами двух взаимодействующих молекул. Константы r_0 и представляют приблизительный молекулярный диаметр и общую сила взаимодействия. Численные значения этих констант будут зависеть от конкретного типа молекулы;

В таблице ниже приведены некоторые значения, полученные путем подгонки модели Леннарда-Джонса к экспериментальным данным для благородных газов при низких плотностях.

	r_0 (Å)	ϵ (eV)
helium	2.65	0.00057
neon	2.76	0.00315
argon	3.44	0.0105

На рисунке 5.2 показан график потенциала Леннард-Джонса. Когда r >> r_0, энергия отрицательна и приближается к нулю пропорционально 1 / r^6 , Это правильное поведение так называемой силы Ван-дер-Ваальса, и может быть выведено из квантовой теории. Когда r <r_0 энергия становится положительной, очень быстро растет, давая сильную силу отталкивания при уменьшении r. Джон Леннард-Джонс предложил использовать закон, чтобы смоделировать это отталкивающее поведение.

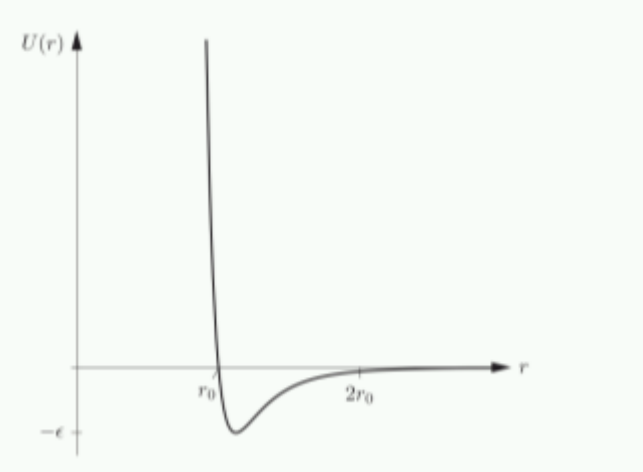


Рисунок 5.2: Функция потенциальной энергии Леннарда-Джонса (уравнение 5.1).

Задание 1. Найдите значение *r* (как функцию r_0), при котором функция Леннарда-Джонса достигает минимума и покажите, его значение в этой точке.

Практическое задание №4

Многие физические системы, такие как молекулы, смоделированные в предыдущей работе , так хаотичны, что движения могут быть случайными. Но в таком случае, почему мы должны рассчитать фактические траектории? Для многих целей мы заменить фиктивное «случайное» поведения на фактическое поведение системы.

Случайность также является свойством квантово-механических измерений, для случая распада ядра. В этой работе нужно написать несколько коротких компьютерных программ, которые используют генерируемые компьютером «случайные» числа для моделирования простых физических систем. Методы, которые используют случайные числа для выбора

среди различных возможных результатов называются методами Монте-Карло по названию знаменитого европейского игорного курорта.

В этом проекте вы научитесь работать в традиционной среде Python. Нет необходимости в 3D-графике или рисовать моделируемые системы в физическом пространстве вообще. Также вам не нужно анимировать моделируемые процессы, чтобы смотреть их поведение во времени. Вместо этого вы будете накапливать данные в списках, а затем выведете результаты, все сразу, используя сводные графики.

Задание 1: Усовершенствуйте вашу программу RandomTest, чтобы получить и распечатать значения 50 случайных чисел (цифр) в диапазон от 0 до 9 включительно. Посчитайте, сколько раз каждая цифра встречается, и запишите результаты вашего подсчета ниже.

Приложение 3. Типовые билеты для проведения промежуточной аттестации

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 1

1. Компилируемые и интерпретируемые языки программирования. Примеры и сравнения.

2. Создать exe-приложение вычисления функции Эйлера ф(n) для любого натурального значения n.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 2

1. Общие сведения о языке Python и особенности его стиля программирования.

2. Создать exe-приложение вычисления НОД(a,b) для любых натуральных a и b.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 3

1. Переменные, значения и их типы. Присваивание значения. Ввод значений с клавиатуры.

2. Создать exe-приложение, осуществляющее частотный анализ введенного текста.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 4

1. Встроенные операции и функции. Основные алгоритмические конструкции языка Python.

2. Создать exe-приложение, осуществляющее преобразование введенного текста по методу Цезаря (сдвиг каждой буквы исходного теста вправо на 3 позиции).

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 5

1. Условный оператор в Python. Множественное ветвление. Условия равенства / неравенства.

2. Создать exe-приложение, которое число из десятичной системы счисления, введенное по запросу с клавиатуры, преобразует в двоичную систему.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 6

1. Циклы и счетчики в Python.

2. Создать exe-приложение, которое двоичное число, введенное по запросу с клавиатуры, в случае правильной записи преобразует в десятичное число.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 7

1. Определение функций. Параметры и аргументы. Вызовы функций. Оператор возврата return.

2. Создать exe-приложение, которое вычисляет ab (modc) для любых натуральных a, b, c.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 8

1. Определение и работа с классами в Python. Методы и параметры класса.

2. Создать exe-приложение, которое вычисляет НОК двух натуральных чисел a и b, по запросу введенных с клавиатуры.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 9

1. Типы последовательностей в Python. Операторы общие для всех типов последовательностей.

2. Создать exe-приложение, которое решает квадратное уравнение ax 2 + bx + c = 0 для любых чисел a, b и c или выводит сообщение, что вещественных значений нет.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 10

1. Списки в Python. Специальные операторы и функции для работы со списками.

2. Создать exe-приложение, которое вычисляет факториал любого натурального числа n.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 11

1. Работа со словарями. Методы словарей. 2. Создать exe-приложение, осуществляющее частотный анализ введенного текста.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 12

1. Вложенные списки. Матрицы.

2. Создать exe-приложение-игру онлайн по типу «камень, ножницы, бумага».

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 13

1. Основные стандартные модули и пакеты в Python. Импортирование модулей.

2. Создать exe-приложение, которое напечатает введенный текст в обратной последовательности (как читалось бы слева направо).

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 14

1. Специализированные модули и приложения в Python.

2. Создать exe-приложение, вычисляющее определитель матрицы 3×3.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 15

1. Создание собственных модулей и их импортирование.

2. Вывести все квадраты натуральных чисел, не превосходящие введенного числа N.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 16

1. Создание независимых exe-приложений в Python.

2. По координатам двух точек, которые вводит пользователь, определить уравнение прямой, проходящей через эти точки.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 17

1. Типы последовательностей в Python. Операторы общие для всех типов последовательностей.

2. Определить, кратно ли заданное число трем; если нет, вывести остаток.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ 18

1. Встроенные операции и функции. Основные алгоритмические конструкции языка Python .

2. Дано число. Найти сумму и произведение его цифр.

Приложение 4. Примерные варианты заданий для проведения контрольных работ

Задание 1. Простейшие арифметические операции

Написать функцию arithmetic, принимающую 3 аргумента: первые 2 - числа, третий - операция, которая должна быть произведена над ними. Если третий аргумент +, сложить их; если —, то вычесть; * — умножить; / — разделить (первое на второе). В остальных случаях вернуть строку "Неизвестная операция".

Задание 2. Високосный год

Написать функцию is_year_leap, принимающую 1 аргумент — год, и возвращающую True, если год високосный, и False иначе.

Задание 3. Квадрат

Написать функцию square, принимающую 1 аргумент — сторону квадрата, и возвращающую 3 значения (с помощью кортежа): периметр квадрата, площадь квадрата и диагональ квадрата.

Задание 4. Времена года

Написать функцию season, принимающую 1 аргумент — номер месяца (от 1 до 12), и возвращающую время года, которому этот месяц принадлежит (зима, весна, лето или осень).

Задание 5. Банковский вклад

Пользователь делает вклад в размере a рублей сроком на years лет под 10% годовых (каждый год размер его вклада увеличивается на 10%. Эти деньги прибавляются к сумме вклада, и на них в следующем году тоже будут проценты).

Написать функцию bank, принимающая аргументы a и years, и возвращающую сумму, которая будет на счету пользователя.

Задание 6. Простые числа

Написать функцию is_prime, принимающую 1 аргумент — число от 0 до 1000, и возвращающую True, если оно простое, и False - иначе.

Задание 7. Правильная дата

Написать функцию date, принимающую 3 аргумента — день, месяц и год. Вернуть True, если такая дата есть в нашем календаре, и False иначе.

Задание 8. XOR-шифрование

Написать функцию XOR_cipher, принимающая 2 аргумента: строку, которую нужно зашифровать, и ключ шифрования, которая возвращает строку, зашифрованную путем применения функции XOR (^) над символами строки с ключом. Написать также функцию XOR_uncipher, которая по зашифрованной строке и ключу восстанавливает исходную строку.

Приложение 5. Виды шкал оценивания

ШКАЛА ОЦЕНИВАНИЯ ПРАКТИЧЕСКИХ РАБОТ
(текущий контроль)

85-100 % - Демонстрирует полное понимание проблемы. Все требования, предъявляемые к заданию, выполнены.

70-84 % - Демонстрирует значительное понимание проблемы. Все требования, предъявляемые к заданию, выполнены.

60-69 % - Демонстрирует частичное понимание проблемы. Большинство требований, предъявляемых к заданию, выполнены.

31-60 % - Демонстрирует небольшое понимание проблемы. Многие требования, предъявляемые к заданию, не выполнены.

0-30 % - Демонстрирует непонимание проблемы и даже не было попытки решить задачу.

ШКАЛА ОЦЕНИВАНИЯ ПИСЬМЕННОГО ОПРОСА
(промежуточный контроль – «ЗНАТЬ»)

Отметкой (9-10 баллов) оценивается ответ, который показывает прочные знания основных понятий информационных технологий, программирования, способов представления информационных моделей, анализа предметной области, этапы разработки программного обеспечения и средства их реализации.

Отметкой (7-8 баллов) оценивается ответ, обнаруживающий прочные знания основных понятий информационных технологий, программирования, способов представления информационных моделей, элементов анализа предметной области, знает этапы разработки программного обеспечения и средства их реализации. Однако допускается одна - две неточности в ответе.

Отметкой (4-6 баллов) оценивается ответ, свидетельствующий в основном о знании информационных технологий, способов представления информационных моделей, элементов программирования. Допускается несколько ошибок в содержании ответа.

Отметкой (2-3 балла) оценивается ответ, обнаруживающий незнание основных понятий информационных технологий, способов представления информационных моделей, элементов разработки программного обеспечения и средств их реализации. Отмечается отсутствие логичности и последовательности в ответе. Допускаются серьезные ошибки в содержании ответа.

Отметкой (0 -1 балл) оценивается ответ, при котором студент демонстрирует непонимание проблемы или нет ответа.

ШКАЛА ОЦЕНИВАНИЯ ПРАКТИЧЕСКИХ ЗАДАНИЙ
(промежуточный контроль – «УМЕТЬ и ВЛАДЕТЬ»)

Отметкой (17-20 баллов) оценивается ответ, при котором студент правильно решает индивидуальные задачи по предмету. · Демонстрирует полное понимание проблемы. Все требования, предъявляемые к заданию, выполнены.

Отметкой (11-16 баллов) оценивается ответ, при котором студент в основном правильно решает индивидуальные задачи по предмету. Демонстрирует значительное понимание проблемы. Большинство требований, предъявляемых к заданию, выполнены.

Отметкой (4-10 баллов) оценивается ответ, при котором студент в основном неправильно решает индивидуальные задачи по предмету; демонстрирует неспособность грамотно формализовать задачу из индивидуального задания. Демонстрирует частичное или небольшое понимание проблемы. Многие требования, предъявляемые к заданию, не выполнены.

Отметкой (0-3 балла) оценивается ответ, при котором студент демонстрирует непонимание проблемы или нет ответа и даже не было попытки решить задачи.

ШКАЛА ОЦЕНИВАНИЯ ОТЧЕТА ПО САМОСТОЯТЕЛЬНОЙ РАБОТЕ

	Наименование показателя	Отметка (в %)
КАЧЕСТВО САМОСТОЯТЕЛЬНОЙ РАБОТЫ		
1	Соответствие содержания работы заданию	0 - 20
2	Грамотность изложения и качество оформления работы	30 - 50
3	Самостоятельность выполнения работы, глубина проработки материала, использование рекомендованной и справочной литературы	0 - 20
4	Обоснованность и доказательность выводов	0 - 10
Общая оценка за выполнение (текущий и рубежный контроль)		Сумма %
КАЧЕСТВО ДОКЛАДА		
1	Соответствие содержания доклада содержанию работы	40 – 60
2	Выполнение основной мысли работы	0 – 20
3	Качество изложения материала	0 – 20
Оценка за доклад (промежуточный контроль)		Сумма %
ОТВЕТЫ НА ДОПОЛНИТЕЛЬНЫЕ ВОПРОСЫ ПО СОДЕРЖАНИЮ РАБОТЫ		
1	Вопрос 1	0 – 25
2	Вопрос 2	0 – 25
3	Вопрос 3	0 – 25
4	Вопрос 4	0 – 25
Оценка за ответы на вопросы (промежуточный контроль)		Сумма %
Общая оценка за промежуточный контроль		Среднее арифм. %

Приложение 6. Технологическая карта дисциплины

Название модулей дисциплины согласно РПД	**Контроль**	**Форма контроля**	**зачетный минимум**	**зачетный максимум**	**график контроля**
Модуль 1					
Основы работы с языком программирования Python	Текущий контроль	Практические работы №1, 2	6	11	27
	Рубежный контроль	Контрольная работа 1	6	11	
Модуль 2					
Последовательности в Python. Модули и пакеты в Python	Текущий контроль	Практические работы №3, 4	7	12	32
	Рубежный контроль	Контрольная работа 2	7	12	
Модуль 3					
Обработка данных физических экспериментов на языке Python	Текущий контроль	Практические работы №5, 6	7	12	36
	Рубежный контроль	Защита отчета по самостоятельной работе	7	12	
ВСЕГО за семестр			40	70	
Промежуточный контроль (ЗачетСОц)			20	30	40
Семестровый рейтинг по дисциплине			60	100	