Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Новосибирский государственный технический университет»
Кафедра вычислительной техники

Паспорт экзамена

по дисциплине «Информатика», 1 семестр

1. Методика оценки

Экзамен проводится в устной форме по билетам. Билет формируется по следующему правилу:

1. Теоретическая часть – в виде 10 тестовых вопросов из разных тем (темы – из списка экзаменационные вопросов, тестовые вопросы выбираются случайным образом из закрытого списка. На каждый тестовый вопрос дается короткий письменный ответ 1-2 абзаца. Тест оценивается 0..2 балла, суммарно 0..20.
2. Задача – выбирается из списка. Оценка 0..20 баллов. Задача реализуется в виде компьютерной программы, исполняемой на тестовых данных.

Таким образом, проверяются результаты обучения по дисциплине, соотнесенные с установленными в программе индикаторами достижения компетенций.

На экзамене преподаватель вправе задавать студенту уточняющие и дополнительные вопросы из общего перечня (п. 4).

Форма экзаменационного билета

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
Факультет АВТФ

Билет № _____
к экзамену по дисциплине «Информатика»

Тестовый вопрос 1 -
Тема: 1. Структура программы и языка программирования. Программа = алгоритм + данные. Структура системы исполнения кода. Особенности Си/Си+

2. Определение компилятора и интерпретатора

Тестовый вопрос 2 -
Тема: 12. Структура данных как система взаимосвязанных переменных и значений. Стек. Представление стека в массиве. Свойства. Использование стека при вызове функции

1. Определение структуры данных

Тестовый вопрос 3 -
Тема: 2. Основы компьютерной архитектуры. Прямо адресуемая память, сегментация, система команд, исполнение программы, поток команд

6. Какие типы команд включает в себя система команд? Как это согласуется с блок-схемой программы?

Тестовый вопрос 4 -
Тема: 23. Идеи (заповеди) структурного программирования – нисходящее, пошаговое, структурное проектирование программы и данных. Последовательность, ветвление и цикл. Модульное программирование.

5. В чем идея структурного проектирования?

Тестовый вопрос 5 -
Тема: 2. Основы компьютерной архитектуры. Прямо адресуемая память, сегментация, система команд, исполнение программы, поток команд

2. Особенности сегмента кода с точки зрения его разделения?

Тестовый вопрос 6 -
Тема: 4. Позиционная система счисления. Представление чисел. Прямое и обратное преобразование из 10СС в СС с другим основанием. Арифметика в СС с другим основанием (сложение, умножение). Особенности двоичной СС.

7. Взять два десятичных числа - дата рождения (2 цифры) и месяц рождения (2 цифры). Цифры 0 заменить на 9. Перевести в 8СС и 16СС. Сложить полученные числа в 8СС и 16СС

Тестовый вопрос 7 -
Тема: 2. Основы компьютерной архитектуры. Прямо адресуемая память, сегментация, система команд, исполнение программы, поток команд

8. Определение потока команд. Как оно связано с архитектурой процессора?

Тестовый вопрос 8 -
Тема: 6. Понятие типа данных и переменной. Определение переменных в Си. Базовые типы данных char,int,long как машинные слова. Особенности типа данных char. Знаковая и беззнаковая формы представления в Си. Представление символов. Представление чисел с плавающей запятой

5. Формула диапазона представления целого со знаком и без знака для n-разрядного машинного слова

Тестовый вопрос 9 -
Тема: 23. Идеи (заповеди) структурного программирования – нисходящее, пошаговое, структурное проектирование программы и данных. Последовательность, ветвление и цикл. Модульное программирование.

6. Как принцип одновременного проектирования алгоритма и данных связан с пошаговым нисходящим проектированием?

Тестовый вопрос 10 -
Тема: 21. Основы анализа программ. “Исторический” и логический анализ программы. “Смысл” выражений и переменных. Стандартные программные контексты (паттерны).

1. Паттерн «место остановки цикла». Вставка в упорядоченную последовательность, 2 последовательных цикла: первый находит место, второй - вставляет (фрагмент кода).

Задача 29. Найти в строке и удалить из нее последовательность повторяющихся символов максимальной длины (например, "abcxxxxxacbyyyyyyyyyyyyyyyyz" - "abcxxxxxacbz").

Утверждаю: зав. кафедрой _____ _____ должность, ФИО
(подпись)
(дата)

2. Критерии оценки результатов обучения, соотнесенных с уровнями освоения индикаторов достижения компетенций

Ответ на экзаменационный билет (тест) засчитывается на **продвинутом** уровне, если студент проводит сравнительный комплексный анализ материала, выявляет проблемы, предлагает механизмы их решения, представляет количественные характеристики определенных процессов, приводит конкретные примеры, не допускает ошибок и способен обосновать выбор метода решения задачи Совокупность результатов обучения по дисциплине (модулю) и соотнесенных с ними индикаторов достижения компетенций не содержит пробелов. Установленные в программе компетенции сформированы в полном объеме. Оценка составляет *от 35 до 40 баллов.*

Ответ на экзаменационный билет (тест) засчитывается на **базовом** уровне, если студент при ответе на вопросы формулирует основные понятия, дает характеристику процессов, явлений, проводит анализ причин, условий, способен представить количественные и качественные характеристики процессов, не допускает существенных ошибок при решении задачи Совокупность результатов обучения по дисциплине (модулю) и соотнесенных с ними индикаторов достижения компетенций содержит несущественные пробелы. Установленные в программе компетенции сформированы на базовом уровне. Оценка составляет *от 34 до 27 баллов.*

Ответ на экзаменационный билет (тест) засчитывается на **пороговом** уровне, если студент при ответе на вопросы дает определение основных понятий, может показать причинно-следственные связи явлений, при решении задачи допускает непринципиальные ошибки, например, вычислительные. Совокупность результатов обучения по дисциплине (модулю) и соотнесенных с ними индикаторов достижения компетенций содержит пробелы. Установленные в программе компетенции сформированы на пороговом уровне. Оценка составляет *от 26 до 20 баллов.*

Ответ на экзаменационный билет (тест) считается **неудовлетворительным** (ниже порогового уровня), если студент при ответе на вопросы не дает определений основных понятий, не способен показать причинно-следственные связи явлений, при решении задачи допускает принципиальные ошибки. Совокупность результатов обучения по дисциплине (модулю) и соотнесенных с ними индикаторов достижения компетенций содержит существенные пробелы. Установленные в программе компетенции не сформированы. Оценка составляет *менее 20 баллов.*

3. Шкала оценки

Экзамен считается сданным, если сумма баллов по всем заданиям билета составляет от 20 до 40 баллов включительно. Сумма менее 20 баллов признается неудовлетворительным результатом промежуточной аттестации по дисциплине.

В общей оценке по дисциплине экзаменационные баллы учитываются в соответствии с правилами балльно-рейтинговой системы, установленными в НГТУ.

4. **Вопросы к** экзамену **по дисциплине «**Информатика**»**

Теоретические вопросы(темы)

1. Структура программы и языка программирования. Программа = алгоритм + данные. Структура системы исполнения кода. Особенности Си/Си++.
2. Основы компьютерной архитектуры. Прямо адресуемая память, сегментация, система команд, исполнение программы, поток команд.
3. Средства сборки программ в ЯП. Модульное программирование: файлы, библиотеки. Трансляция и компоновка программы. Функция, модуль (файл), проект. Объектный модуль (ОМ), точки входа, внешние ссылки. Компоновка программного модуля из ОМ и библиотек. Программные интерфейсы данных и функций. Заголовочные файлы.
4. Позиционная система счисления. Представление чисел. Прямое и обратное преобразование из 10СС в СС с другим основанием. Арифметика в СС с другим основанием (сложение, умножение). Особенности двоичной СС.
5. Единицы представления информации в компьютере: бит, байт, машинное слово. Двоичная и шестнадцатеричная системы счисления и их использование в компьютере. Оценка информационной емкости двоичного числа. Представление целого без знака в машинном слове.
6. Понятие типа данных и переменной. Определение переменных в Си. Базовые типы данных char,int,long как машинные слова. Особенности типа данных char. Знаковая и беззнаковая формы представления в Си. Представление символов. Представление чисел с плавающей запятой.
7. Массивы: особенности работы, инициализация. Массивы как формальные параметры функций. Связь массивов и указателей. Адресная арифметика.
8. Выражения и операции (обзор и классификация): арифметические, сравнения, логические, машинно-ориентированные, присваивания, адресные, выделения составляющего типа данных. Особенности выполнения операций на Си (совместимость, приоритеты, направление выполнения, действие и результат). Особенности выполнения арифметических операций и операций присваивания. Операция "запятая". Операции сравнения, логические операции.
9. Битовые операции и маски. Применение поразрядных операций для установки, сброса и проверки битов. Использование битовых масок для эффективного хранения флагов и управления аппаратурой.
10. Составные типы данных: структуры (struct), объединения (union), перечисления (enum). Битовые поля. Выравнивание структур в памяти. Использование typedef для создания псевдонимов типов.
11. Функции. Формальные и фактические параметры. Способы передачи параметров -- по значению и по ссылке. Результат функции. Локальные и глобальные (автоматические и внешние) переменные. Функция как основа модульного программирования.
12. Модель памяти процесса. Сегменты памяти (стек, куча, данные, код). Размещение переменных в памяти. Стек вызова функций: кадр стека, адрес возврата, локальные переменные. Анализ стека с использованием отладчика.
13. Динамическое управление памятью. Функции malloc, calloc, realloc, free. Ошибки работы с динамической памятью: утечки памяти, двойное освобождение,

использование после освобождения. Инструменты поиска утечек (Valgrind, AddressSanitizer).

14. Циклические программы. Виды циклов. Итерационный цикл. Рекуррентные последовательности и итерационные циклы. Программа вычисления корня функции. Программа вычисления суммы степенного ряда.
15. Строки в Си. Представление строки как массива символов с нулевым терминатором. Стандартные функции библиотеки string.h: strlen, strcpy, strcat, strcmp, strstr, strchr. Посимвольная и пословная обработка. Безопасные альтернативы (strncpy, strncat, snprintf).
16. Преобразования целого и вещественного числа из внешней формы во внутреннюю и обратно.
17. Работа с текстовыми файлами в stdio. Функции открытия/закрытия файла, посимвольный, построчный и форматированный ввод/вывод. Стандартные потоки. (Примеры).
18. Трудоемкость алгоритмов. Время выполнения и трудоемкость. Оценочный характер трудоемкости. Виды функций трудоемкости, чувствительность к данным. Программные конструкции, определяющие различные виды трудоемкости.
19. Сортировка и поиск. Понятие записи и ключа. Линейный и двоичный поиск. Трудоемкость алгоритмов сортировки и поиска.
20. Классификация сортировок: выбор, вставка, обмен, подсчет, разделение, слияние. Идеи. Трудоемкость. (по тестам к защите л.р. Сортировки)
21. Основы анализа программ. "Исторический" и логический анализ программы. "Смысл" выражений и переменных. Стандартные программные контексты (паттерны). (Показать на примере)
22. Жизненный цикл разработки программы. Образная модель. Предварительный сбор фактов. Технология программирования как «выстраивание» текста программы. «Историческое», структурное и «грязное» программирование (показать на примерах).
23. Идеи (заповеди) структурного программирования -- нисходящее, пошаговое, структурное проектирование программы и данных. Последовательность, ветвление и цикл. Модульное программирование (показать на примерах).
24. Динамические структуры данных. Односвязный и двусвязный список. Стек и очередь как частные случаи списка. Бинарное дерево поиска. Стандартные операции над динамическими структурами. Сравнение трудоемкости операций.
25. Архитектура, протоколы, сервисы Интернет, основы сетевой безопасности и источники и принципы действия угроз компьютерной безопасности.
26. Библиографический поиск и поиск в интернете. Правовые основы информационной безопасности и принципы защиты авторского права на программные продукты.
27. Модульная организация программ. Заголовочные и файлы реализации. Раздельная компиляция и компоновка. Создание статических и динамических библиотек. Сборка программ с использованием библиотек. Механизм динамической загрузки (dlopen, dlsym).

Требования к содержанию и оформлению задачи

1. Задача оформляется в виде функции с передачей всех входных и выходных данных через формальные параметры и результат, размерность записываемых данных

2. Задача, работающая со строками, использует в качестве параметра массив char[], либо указатель char*
3. Для функции привести 2-3 примера вызова из main со статическими входными параметрами – тест. Программа запускается от одного клика.
4. Программа должна компилироваться, запускаться и выводить результат (не зависать и не падать) без вмешательства преподавателя. Любое вмешательство или исправление снижает оценку ответа.
5. По требованию преподавателя в алгоритм и интерфейс функции должны вноситься оперативные изменения, этим оценивается уровень навыков.

Задачи

1. Найти наименьшее общее кратное для всех элементов массива - минимальное число, которое делится на все элементы массива без остатка.
2. Написать функцию проверки, является ли заданное число простым. С ее помощью написать функцию поиска простых чисел в диапазоне 1000-2000, две любые части которого - также простые (например, 1997, 1-997, 19-97, 199-7).
3. Сформировать массив простых чисел в диапазоне от 2 до заданного. Очередное простое число определяется попыткой деления нацело числа на все уже накопленные простые числа.
4. Любая целочисленная денежная сумма n>7 может быть выдана без сдачи "трешками" и "пятерками". Программа находит эти числа для заданного n.
5. На заданном интервале найти все числа, цифры которых находятся в строго возрастающем порядке (например, 532).
6. Число Армстронга -- такое число из k цифр, для которого сумма k-х степеней его цифр равна самому этому числу, например $153=1^3+5^3+3^3$. Найти все трехзначные числа Армстронга.
7. Найти все двухзначные (трехзначные) числа, которые совпадают с последними цифрами своих квадратов, используя битовые операции для проверки (например, $25^2=625$, $76^2=5676$).
8. Разработать функцию, которая с использованием динамического массива находит все совершенные числа в заданном интервале (число равно сумме всех своих делителей, например, 6=1+2+3). Возвращать массив найденных чисел.
9. Найти все числа в заданном диапазоне, которые делятся на любую из своих цифр. Реализовать с использованием битовых масок для проверки делимости.
10. Разработать программу, которая с использованием стека (реализованного на массиве) проверяет баланс скобок в строке (круглые, квадратные, фигурные).
10. Разложить число на простые множители (например 36=3*3*2*2). Результат -- последовательность множителей в массиве, ограниченная 0.
11. Определить в массиве максимальную длину последовательности расположенных подряд возрастающих значений и возвратить индекс ее начала и длину.
12. Подсчитать количество слов в строке.
13. Найти в строке слово минимальной длины и возвратить индекс его начала. Реализовать с использованием функций из string.h.
14. С использованием односвязного списка реализовать стек. Написать функции push, pop, top. Продемонстрировать работу на примере обращения арифметического выражения.
15. С использованием двусвязного списка реализовать очередь. Написать функции enqueue, dequeue, front. Продемонстрировать работу на примере моделирования очереди задач.
17. "Перевернуть" в строке все слова с использованием указателей и адресной арифметики (без использования дополнительного массива). Например: "Жили были дед и баба" → "илиЖ илиб дед и абаб".
16. Преобразовать целое из внутренней формы во внешнюю в шестнадцатеричной СС.
17. Преобразовать число в шестнадцатеричной СС из внешней формы во внутреннюю.

18. Разработать программу, которая с использованием динамической памяти создает односвязный список, заполняет его числами из файла, сортирует и записывает результат обратно в файл.
19. Найти в строке два одинаковых фрагмента, не содержащих пробелы и имеющих максимальную длину, и возвратить индексы их начала. Использовать указатели для обхода строки.
20. Разработать программу для работы с двоичным файлом: запись и чтение массива структур фиксированной длины. Структура содержит поля: целое, вещественное и строку.
23. Разработать программу для работы с текстовым файлом: чтение данных в формате "ключ=значение", сохранение в динамический массив структур, поиск по ключу.
21. Заменить в строке все восьмеричные целые константы на символы с соответствующими кодами (например, xxx101yyy102zzz на xxx Ayyy102zzz).
22. Найти слово, начинающееся с самой младшей латинской буквы, и возвратить индекс его начала.
23. С использованием динамических структур данных (список) реализовать хранение и обработку записей о студентах (ФИО, группа, оценка). Реализовать операции: добавление, удаление, поиск, вывод всех записей.
28. Заменить в строке все большие латинские буквы на соответствующие им шестнадцатеричные коды с использованием битовых операций (например, A на 0x41, использовать 2 цифры для представления байта).
28. Разработать программу, которая с использованием бинарного дерева поиска выполняет вставку, удаление и поиск элементов. Продемонстрировать работу на примере чисел из файла.
24. Найти в строке и удалить из нее последовательность повторяющихся символов максимальной длины (например, "abcxxxxxacbyyyyyyyyyyyyyyyyz" → "abcxxxxxacbz").
29. Разработать программу, которая с использованием стека (на основе динамического списка) выполняет преобразование инфиксной записи в постфиксную (обратную польскую запись).
30. Найти все вхождения подстроки в строке с использованием указателей. Строка и подстрока заданы в массивах символов. Результат -- массив, заполненный индексами начала подстроки в строке, последовательность ограниченна -1.
31. Разработать программу для работы с динамической структурой "очередь с приоритетом" на основе списка. Реализовать операции вставки с приоритетом и извлечения максимального приоритета.

Сортировки реализовать в полном соответствии с требованиями варианта.

1. Сортировка выбором. Выбирается минимальный элемент в массиве и запоминается. Затем удаляется, а все последующие за ним элементы до конца массива сдвигаются на один влево. Сам элемент заносится на освободившуюся последнюю позицию.

2. Сортировка выбором. Выбирается минимальный элемент в массиве и запоминается. Затем удаляется, а все последующие за ним элементы массива сдвигаются на один влево. Сам элемент переносится в новый массив.

3. Сортировка выбором. Выбирается минимальный элемент в массиве и запоминается. Затем на его место записывается «очень большое число», например, максимальное +1. Сам элемент переносится в новый массив.

4. Сортировка выбором. Выбирается максимальный элемент из оставшихся и меняется с последним из них (обратить внимание на диапазон неупорядоченной части).

5. Сортировка выбором. Выбирается минимальный элемент из оставшихся и меняется с первым из них (обратить внимание на диапазон неупорядоченной части).

6. Сортировка подсчетом. Выходной массив заполняется значениями “-1”. Затем для каждого элемента определяется его место в выходном массиве путем подсчета количества элементов строго меньших данного. Естественно, что все одинаковые элементы попадают на одну позицию, за которой следует ряд значений “-1”. После чего оставшиеся в выходном массиве позиции со значением “-1” заполняются копией предыдущего значения.

7. Сортировка вставками. Берется очередной элемент и извлекается из массива. Затем от начала массива ищется первый элемент, больший данного. Все элементы, от найденного до очередного сдвигаются на один вправо и на освободившееся место помещается очередной элемент. (Поиск места включения от начала упорядоченной части).

8. Сортировка вставками («всплытием»). Берется предпоследний элемент и меняется со следующим, пока не закончится массив и пока следующий будет меньше текущего. Затем берется предыдущий и т.д..

9. Сортировка вставками. Берется очередной элемент массива. Затем от начала выходного массива ищется первый элемент, больший данного. Все элементы, от найденного до последнего сдвигаются на один вправо и на освободившееся место помещается очередной элемент.

10. Сортировка выбором. Выбирается минимальный элемент в массиве и переносится в новый массив. Затем на его место записывается последний элемент исходного массива.