Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Новосибирский государственный технический университет»
Кафедра автоматики
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: один теоретический вопрос из списка и одна задача из списка. Задача реализуется в виде компьютерной программы, исполняемой на тестовых данных.

Форма экзаменационного билета

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
Факультет АВТФ

Билет № _____
к экзамену по дисциплине «Информатика»

1.Теоретический вопрос. Выражения и операторы. Роль ";" как ограничителя. Классификации управляющей логики программы - последовательность, условие, цикл, переход. Основные операторы Си: if, while, do-while, for, switch, break, continue, return, goto: классификация, особенности синтаксиса и выполнения

2. Задача. Сформировать массив простых чисел в диапазоне от 2 до заданного. Очередное простое число определяется попыткой деления нацело числа на все уже накопленные простые числа.

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

2. Критерии оценки

- Ответ на экзаменационный билет (тест) считается **неудовлетворительным**, если студент при ответе на вопросы не дает определений основных понятий, не написал текст программы для задачи, оценка составляет *<25% базовой.*
- Ответ на экзаменационный билет (тест) засчитывается на **пороговом** уровне, если студент при ответе на вопросы дает определение основных понятий, но не раскрывает их содержимое, не может привести примеров использования конструкций языка, имеется текст программы, но она не работает или дает неправильный результат, оценка составляет 25-*50% базовой.*
- Ответ на экзаменационный билет (тест) билет засчитывается на **базовом** уровне, если студент при ответе на вопросы формулирует основные понятия, раскрывает их

содержимое, анализирует примеры использования, но не объясняет отдельные особенности применения, программа компилируется и выполняется, но студент не может внести в нее изменения по требованию преподавателя, оценка составляет *50-75% базовой*..

- Ответ на экзаменационный билет (тест) билет засчитывается на **продвинутом** уровне, если студент студент выполняет все перечисленные выше требования, оценка составляет *75-100% базовой*..

3. Шкала оценки

В общей оценке по дисциплине экзаменационные баллы учитываются в соответствии с правилами балльно-рейтинговой системы, приведенными в рабочей программе дисциплины, т.е. 40 баллов за экзамен.

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

Теоретические вопросы

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.
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. Преобразование базовых типов данных в выражениях: действия, порядок. Явные и неявное преобразования.
10. Выражения и операторы. Роль ";" как ограничителя. Классификации управляющей логики программы - последовательность, условие, цикл, переход. Основные операторы

Си: if, while, do-while, for, switch, break, continue, return, goto: классификация, особенности синтаксиса и выполнения.

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. Структура данных как система взаимосвязанных переменных и значений. Стек. Представление стека в массиве. Свойства. Использование стека при вызове функции.
13. Структура данных как система взаимосвязанных переменных и значений. Последовательность. Стек и очередь. Свойства. Представление очереди в массиве.
14. Циклические программы. Виды циклов. Итерационный цикл. Рекуррентные последовательности и итерационные циклы. Программа вычисления корня функции. Программа вычисления суммы степенного ряда.
15. Работа со строками. Представление строки в Си. Строка и массив символов. Поиск в строке. Посимвольная и пословная обработка. Примеры.
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. (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. Время жизни и область действия переменных и функций. Определения и объявления. Глобальные, локальные переменные и их свойства (показать на примерах).
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. Информация. Определения. Меры. Представление данных в компьютере. Бит (разряд), байт, машинное слово. Представление целых со знаком и без знака, вещественных чисел и символом. Представление мультимедийных данных - аудио, видео,

изображение. Растровая форма, сжатие без потери и с потерей качества. Битрейд. Избыточность, архивирование.

Задачи

Задачи оформляются в виде функций с передачей всех входных и выходных данных через формальные параметры и результат (глобальные переменные использовать запрещается). В main пишется несколько вызовов функции на статических данных, с выводом результатов, демонстрирующих работоспособность программы.

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, например, $25^2=625$, $76^2=5676$.

8. Число называется совершенным, если оно равно сумме всех своих делителей, например, 6=1+2+3, 28=1+2+4+7+14. Найти все совершенные числа в заданном интервале.

9. Найти все числа в заданном диапазоне, которые делятся на любую из своих цифр.

10. Найти числа в диапазоне 100-10000, для которых куб суммы цифр равен значению самого числа (например, 512 - 5+1+2=8).

11. Разложить число на простые множители (например 36=3*3*2*2). Результат – последовательность множителей в массиве, ограниченная 0.

12. Определить в массиве максимальную длину последовательности расположенных подряд возрастающих значений и возвратить индекс ее начала и длину.

13. Подсчитать количество слов в строке

14. Найти в строке слово минимальной длины и возвратить индекс его начала.

15. Возвратить индекс начала фрагмента, симметричного относительно центрального символа (например, "abcba"), и имеющего максимальную длину. Длину фрагмента возвратить по ссылке.

16. Возвратить массив индексов начала фрагмента, симметричного относительно центрального символа (например, "abcba"), и имеющих длину >=5.

17. "Перевернуть" в строке все слова. (Например: "Жили были дед и баба" - "илиЖ илиб дед и абаб").

18. Преобразовать целое из внутренней формы во внешнюю в шестнадцатеричной СС.

19. Преобразовать число в шестнадцатеричной СС из внешней формы во внутреннюю.

20. Преобразовать дробную часть переменной типа double во внешнюю форму представления (строку символов).

21. Найти в строке два одинаковых фрагмента, не содержащих пробелы и имеющих максимальную длину и возвратить индексы их начала

22. В строке находится символ “точка” и символы-цифры дробной части числа. Преобразовать во внутреннюю форму представления (переменную типа double).

23. В строке, содержащей абзац текста, найти концы предложений, обозначенный символом "точка". В следующих за ними словах первую строчную букву заменить на прописную. Между словами количество пробелов может быть любым.

24. Заменить в строке все восьмеричные целые константы на символы с соответствующими кодами, (например, xxx101yyy102zzz на xxx Ayyy102zzz).

25. Найти слово, начинающееся с самой младшей латинской буквы и возвратить индекс его начала.

26. Удалить из строки комментарии вида "/* ... */". Игнорировать вложенные комментарии.

27. Заменить в строке все большие латинские буквы на соответствующие им шестнадцатеричные коды (например, A на 0x41, в константе использовать 2 цифры для представления байта).

28. Заменить в строке все целые константы из любого количества цифр соответствующим повторением следующего за ними символа (например "abc5xacb15y" - " abcxxxxxacbyyyyyyyyyyyyyyy ").

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

30. Найти в строке наиболее часто встречающийся символ и заменить его на пробел.

31. Найти все вхождения подстроки в строке. Строка и подстрока заданы в массивах символов. Результат – массив, заполненный индексами начала подстроки в строке,

последовательность ограниченна -1.

32. Найти в строке самую внутреннюю пару скобок и удалить содержащийся между ними текст

Сортировки реализовать в полном соответствии с требованиями варианта.

33. 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.

34. 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.

35. 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.

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

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

38. 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.

39. 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).

40. 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..

41. 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.

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