Практикум по циклу дисциплин «Информатика»

часть II

Институт №8 «Компьютерные науки и прикладная математика»
Кафедра 806
I курс, II семестр 2021/22 учебного года
Направления ПМИ, ПМ и ФИИТ

—Устройство для преобразования стека в очередь?
Автомат Калашникова!
Юмор студентов ВМК

«Инструментальные средства UNIX, алгоритмы и структуры данных»

График выполнения заданий курсового проекта и лабораторных работ

Практикум Лаб. работа задание № №	Макс. оценка	Сдача зачета	Отладка	Оформление отчета и защита работы
20, 21, 22	5	4 неделя	6 неделя	8 неделя
	4	5 неделя	7 неделя	9 неделя
	3	6 неделя	8 неделя	10 неделя
VI, VII	5	6 неделя	8 неделя	10 неделя
	4	7 неделя	9 неделя	11 неделя
	3	8 неделя	10 неделя	12 неделя
VIII 23, 24	5	8 неделя	10 неделя	12 неделя
	4	9 неделя	11 неделя	13 неделя
	3	10 неделя	12 неделя	14 неделя
IX 25**,** 26	5	10 неделя	12 неделя	14 неделя
	4	11 неделя	13 неделя	15 неделя
	3	12 неделя	14 неделя	16 неделя
Отчет по всем заданиям и по лабораторным работам (в целом)	5	15 неделя		
	4	16 неделя		
	3	**17 неделя**		

Учебным планом II семестра по дисциплинам цикла информатики предусмотрены: экзамен или рейтинговый зачёт по теоретическому курсу, и зачёт по практикуму также в балльно-рейтинговой форме, выставляемый на основании оценок за лабораторные работы и задания практикума, результатов сдачи *промежуточных зачётов VI–IX* (суммарный объём лекций 68-102 час.). Возможен также курсовой проект (трудоёмкость 72-108 час.), оцениваемый в пятибалльной системе, включая беллетристическое задание (эссе, реферат, ...) по одной из тем теоретического или практического курса (в составе отчёта по КП), необязательное для проектов, оцениваемых удовлетворительно, и лабораторные работы (68-102 час.).

Тестирование и протоколирование программ осуществляются на 64-битном сервере неинтеловской архитектуры DEC AXP (Alpha) в среде UNIX NetBSD с интерпретатором команд Bash и системой программирования GNU CC с терминалов и рабочих станций UNIX, за исключением заданий, оцениваемых на удовлетворительно, которые допускается тестировать в лабораторной среде на платформе Intel. Контрольные распечатки выполняются исключительно на лабораторном принтере LA-120 (при наличии). Для отладки могут использоваться домашние компьютеры студентов и лабораторные рабочие станции интеловской архитектуры.

Плановое время индивидуальной работы за терминалом UNIX - 102 часа (17 раз по 6 академических часов по расписанию в 438 или 440 ГУК «Б») в период с 9.02.2022 г. по 08.06.2022 г., причем *последние недели следует отвести для оформления и сдачи отчётов*. Для самостоятельной работы по заданиям курсового проекта в классе предусмотрены дополнительные места сверх средней численности групп, доступные не менее 30 часов в неделю во время лабораторных работ других групп потока.

Во II семестре необходимо помнить, что *в отличие от I семестра* преподаватели уходят в **отпуск** сразу же после окончания сессии. Поэтому не следует рассчитывать на консультации и приём задолженностей в период летних каникул. Также следует иметь в виду, что весенняя сессия – переводная, заканчивается после сдачи всех зачётов и экзаменов приказом о переводе на следующий курс, который должен быть издан до 26 сентября. Т.е., времени на пересдачи меньше, чем было в I семестре.

Правила оформления отчетов прежние. По согласованию с преподавателем возможно оформление отчета в электронном виде на CD-R (CD-RW, DVD-RAM/±R/±RW) в одном из форматов MS Word, HTML, TeX, XML, SGML, PDF, PostScript. При оформлении отчета в электронном виде преподавателю предоставляются титульный лист, бланк задания, носитель с отчётом и листы с подписанными листингами.

Применение систем автоматического тестирования и электронной подписи также оставляется на усмотрение преподавателя, если иное не предусмотрено правилами.

Задание VI. Обработка последовательной файловой структуры на языке Си

Разработать последовательную структуру данных для представления простейшей базы данных на файлах в СП Си в соответствии с заданным вариантом. Составить программу генерации внешнего нетекстового файла заданной структуры, содержащего представительный набор записей (не менее 20). Распечатать содержимое сгенерированного файла в виде таблицы и выполнить над ним заданное действие для 3–5 значений параметров запроса ***p*** и распечатать результат.

Действие по выборке данных из файла оформить в виде *отдельной программы* с параметрами запроса, вводимыми из стандартного входного текстового файла, или получаемыми из командной строки UNIX. Второй способ задания параметров обязателен для работ, оцениваемых на хорошо и отлично. Параметры задаются с помощью ключей –f (распечатка файла) или –p <parameter> (параметры конкретного варианта задания). Получение параметров из командной строки производится с помощью стандартных библиотечных функций argc и argv.

Структуры данных и константы, совместно используемые программами, следует вынести в отдельный заголовочный файл.

В процессе отладки и тестирования рекомендуется использовать команды обработки текстовых файлов ОС UNIX и переадресацию ввода-вывода. Сгенерированные и отформатированные тестовые данные необходимо заранее поместить в текстовые файлы и распечатывать при протоколировании. Рекомендуется подобрать реальные или правдоподобные тестовые данные. Число наборов тестовых данных должно быть не менее трёх. Имя файла с бинарными данными является обязательным параметром второй программы.

Отчёт должен содержать оценку пространственной и временной сложности использованного алгоритма. В состав отчета также рекомендуется включить графическую иллюстрацию структуры файла и запроса на выборку.

В качестве дополнительного задания (принимается в качестве идеи решения задачи и способа тестирования):

– описать структуру файла как реляционную таблицу и сформулировать действие в виде запроса на структурированном языке запросов SQL, или на Прологе [10, 11]. Так, задание варианта 48 может быть специфицировано на языке SQL следующим образом:

Описание таблицы: CREATE TABLE ABIT (SURNAME CHAR (80), INITIALS CHAR(2), MATH INTEGER, PHYS INTEGER, LIT LOGICAL);
Запрос: SELECT SURNAME, INITIALS FROM ABIT WHERE (LIT=TRUE) AND (MATH+PHYS> (SELECT AVG(MATH+PHYS) FROM ABIT WHERE LIT=TRUE));

– добавить проверку правильности работы процедуры запроса в протоколе, путем сравнения её результатов с результатами, получаемыми из исходных текстовых файлов командами UNIX.

Варианты заданий

Содержимое и структура файла

1 –11. Сведения о составе комплектующих личных ПЭВМ в студенческой группе: фамилия владельца, число и тип процессоров, объём памяти, тип видеоконтроллера (встроенный, внешний, видео-шина) и объём видеопамяти, тип (SAS/SATA), число и ёмкость винчестеров, количество интегрированных контроллеров и внешних (периферийных) устройств, операционная система.

12 – 21. Информация об успеваемости студентов данной группы по всем предметам: фамилия, инициалы, пол, номер группы, отметки по экзаменам и зачетам.

22 – 31. Сведения о вступительных экзаменах абитуриентов: фамилия, инициалы, пол, номер школы, наличие медали, оценки в баллах и зачет/незачет по сочинению.

32 – 39. Информация о пассажирах аэропорта: фамилия, инициалы, количество вещей, общий вес вещей, пункт назначения, время вылета, наличие пересадок, сведения о детях.

40 – 47. Общая информация о выпускниках школы студента: фамилия, инициалы, пол, номер класса, буква класса, в каком ВУЗе учится, где работает, в каком полку служит и т.п.

По усмотрению преподавателя задачи могут быть сформулированы, в соответствии с номером группы, для сотрудников фирмы (1), преподавателей кафедры (2), больных в больнице (3), жильцов дома (4), рейтинговых таблиц спортсменов (5), хит-парадов (6), осужденных в местах заключения (7), залогодателей ломбарда (8), клиентов службы знакомств (9), покойников на кладбище (10), покупателей интернет-магазина (11), абонентов телефонных компаний (12), владельцев автомобилей (13) и т.д.

Тестовые данные не должны нарушать действующее законодательство о персональных данных.

Действия (* обозначены более сложные задания)**:**

1. Найти всех владельцев двухпроцессорных компьютеров, имеющих не более ***p*** внешних устройств.
2.* Напечатать список однофамильцев, имеющих однотипные компьютеры.
3.* Распечатать типичные конфигурации компьютеров в группе (более ***p*** владельцев).
4. Отпечатать список студентов, компьютеры которых нуждаются в апгрейде (более ***p*** устройств).
5.* Для всех студентов, имеющих более одного компьютера, распечатать сведения о самом мощном из них.
6. Распечатать сведения обо всех компьютерах-серверах и рабочих станциях.
7. Составить аннотированный список неукомплектованных компьютеров (некомплект – ***p*** устройств).
8. Составить список мультимедийных компьютеров и бездисковых рабочих станций.
9. Составить список плохо сконфигурированных компьютеров.
10. Составить список компьютеров с фирменными комплектующими.
11. Перечислить все компьютеры студентов группы, платформа которых отлична от WINTEL.
12. Выяснить, сколько студенток группы ***p*** получают стипендию.
13. Выяснить, сколько студенток группы ***p*** имеют ровно одну пятёрку.
14. Выяснить, сколько студентов группы ***p*** имеют больше двух троек.
15. Напечатать список потенциальных стипендиатов – студентов, у которых одна тройка, а все остальные оценки четвёрки и пятёрки или все пятёрки и одна четвёрка.
16. Найти фамилии лучших студенток курса (не имеющих отметок ниже четырех и по сумме баллов, не уступающих другим студентам своей группы).
17. Выяснить, в какой группе студентки имеют максимальный средний балл.
18.* Выяснить, в какой группе разность между максимальным и минимальным средним баллом студентов максимальна.
19.* Выяснить, в какой группе учится максимальное число студентов с минимальным на курсе средним баллом.
20.* Выяснить, в какой группе учится максимальное число студенток с максимальным на курсе средним баллом.
21.* Напечатать список ***p*** лучших студентов курса (с наивысшими средними баллами).
22. Найти абитуриентов-медалистов, не набравших проходной балл ***p***.
23. Найти абитуриентов-медалистов, получивших неудовлетворительную оценку по математике.
24. Найти абитуриентов, имеющих заданную сумму баллов ***p***.
25. Найти абитуриентов, имеющих сумму баллов от p_1 до p_2.
26. Найти абитуриентов-немедалистов, суммарный балл которых выше среднего.
27. Найти абитуриенток, получивших по двум предметам одинаковые оценки.
28. Найти абитуриенток, имеющих по всем предметам разные оценки.
29. Найти абитуриентов, получивших максимальную оценку по одному предмету, но не набравших проходного балла ***p***.
30. Найти абитуриенток, получивших одинаковые оценки по всем предметам, но не набравшим проходного балла ***p***.
31.* Найти абитуриентов, имеющих полупроходной балл, при наличии ***p*** мест в плане приёма.
32. Найти пассажиров, вес багажа которых отличается от максимального веса менее чем на ***p*** кг.
33.* Найти пассажира, средний вес вещей багажа которого отличается не более чем на ***p*** кг от среднего веса вещей пассажиров для каждого рейса.
34. Найти пассажиров, имеющих более ***p*** вещей.
35. Найти пассажиров, число вещей которых превосходит среднее число вещей не менее, чем на ***p*** штук.
36.* Определить, имеются ли два пассажира, багаж которых совпадает по числу вещей и различается по весу не более чем на ***p*** кг.
37. Выяснить, имеется ли пассажир, багаж которого состоит из p_1 вещей весом не менее p_2 кг.
38.* Дать сведения о пассажирах, число вещей которых не меньше, чем в любом другом багаже, а вес вещей не больше, чем в любом другом багаже с этим же числом вещей.
39. Выяснить, имеется ли пассажир, багаж которого превышает багаж каждого из остальных пассажиров и по числу вещей и по весу.
40.* Выяснить, имеются ли в школе однофамильцы.
41.* Выяснить, имеются ли однофамильцы в каких-либо параллельных классах.
42.* Выяснить, имеются ли однофамильцы в каком-нибудь классе.
43.* Выяснить, в каком классе учится максимальное число учениц.
44. Выяснить, на сколько учеников в ***p***-х классах школы больше, чем в одиннадцатых классах.
45.* Найти среднее число учениц в классах школы.
46.* Найти классы, в которых число учеников больше числа учениц.
47. Найти классы, выпускники которых либо поступили в вузы, либо призваны на военную службу.

Пример

- Тип данных: имя и фамилия больного, температура.
- Задание: вывести имена и фамилии больных, температура которых ниже средней по больнице.
- Проект состоит из трех файлов: person.h, persons_dump.c, cool_persons.c

person.h:

```
#ifndef __person_h__
#define __person_h__

typedef struct {
  char name[50];
  int temp;
} person;

#endif
```

persons_dump.c

```
#include <stdio.h>
#include <string.h>
#include <errno.h>

#include "person.h"

void usage()
{
  printf("Usage: program filename\n");
}

int readperson(person *p)
{
  return scanf("%[^\t]\t%d\n", p->name, &p->temp) == 2;
}

int main(int argc, char* argv[])
{
  if (argc != 2) {
    usage();
    return 1;
  }
  person p;
  FILE *out = fopen(argv[1], "w");
  if (!out) {
    perror("Can't open file");
    return 2;
  }
  while (readperson(&p))
    fwrite(&p, sizeof(p), 1, out);
  return 0;
}
```

cool_persons.c

```
#include <stdio.h>
  #include <stdlib.h>

  #include "person.h"

  /*
  Программа просматривает данные бинарного файла пациентов больницы
  и выводит имена и фамилии больных, температура которых меньше средней
  по больнице
  */

 void usage()
 {
   printf("Usage: program filename\n");
 }

 int main(int argc, char* argv[])
 {
   if (argc != 2) {
     usage();
     return 1;
   }
   person p;
   FILE *in = fopen(argv[1], "r");
   if (!in) {
     perror("Can't open file");
     return 2;
   }
   int temp_sum = 0;
   int n = 0;
   while (fread(&p, sizeof(p), 1, in) == 1) {
     temp_sum += p.temp;
     ++n;
   }
   fseek(in, 0, SEEK_SET);
   if (n == 0) {
     printf("No people, average temperature is not defined\n");
     return 3;
   }
   double avg = (double)temp_sum / n;
   while (fread(&p, sizeof(p), 1, in) == 1)
     if (p.temp < avg)
      printf("%s\n", p.name);
  return 0;
}
```

Вопросы для сдачи зачёта к ЛР № 20 и заданию VI …

… приведены в описании ЛР № 20

Литература к заданию VI

1. *Кристиан К. Введение в операционную систему* UNIX. *–М.: Финансы и статистика, 1985.*
2. *Беляков И.Н., Рабовер Ю.И., Фридман А.Л. Мобильная операционная система: Справочник. –М.: Радио и связь, 1991.*
3. *Баурн С. Операционная система UNIX. –М.: Мир, 1986.*

4. *Зайцев В.Е. и др. CD-хрестоматия по курсу информатики. –М.: МАИ, 1997-2004.*
5. *Дейт К. Дж. Введение в системы баз данных. – М.: Наука, 1981.*
6. *Каймин В.А., Титов В.К., и др. Информатика. Учебное пособие и сборник задач с решениями (для школьников). — М.: Бридж, 1994.*

Вопросы для самостоятельного изучения к заданию VII и сдачи промежуточного зачёта VII

РАЗРЕЖЕННЫЕ МАТРИЦЫ

1. Представление массивов в памяти ЭВМ.
2. Адресация элементов массивов и ее использование для представления структур данных.
3. Передача параметров-массивов и параметров-записей.
4. Ошибки адресации массивов и их последствия при выполнении программ в операционных системах с защитой памяти и без неё.
5. Приемы обработки и ввода/вывода массивов на скалярных ЭВМ.
6. Представление обычных и вариантных записей (структур) в памяти ЭВМ.
7. Приемы обработки и ввода/вывода записей.
8. Разреженные матрицы. Их представление в памяти ЭВМ.
9. Особенности хранения в памяти ЭВМ треугольных, симметричных и квазидиагональных матриц.
10. Приемы хранения и обработки разреженных матриц на языке Си.

Задание VII. Разреженные матрицы.

Составить программу на языке Си с процедурами и/или функциями для обработки *прямоугольных* разреженных матриц с элементами целого (группы 6, 8), вещественного (группы 2-5), или комплексного (группы 1, 7) типов, которая:

1. вводит матрицы различного размера, представленные во входном текстовом файле в обычном формате (по строкам), с одновременным размещением ненулевых элементов в разреженной матрице в соответствии с заданной схемой;
2. печатает введенные матрицы во внутреннем представлении согласно заданной схеме размещения **и** в обычном (естественном) виде;
3. выполняет необходимые преобразования разреженных матриц (или вычисления над ними) путем обращения к соответствующим процедурам и/или функциям;
4. печатает результат преобразования (вычисления) согласно заданной схеме размещения и в обычном виде.

В процедурах и функциях предусмотреть проверки и печать сообщений в случаях ошибок в задании параметров. Для отладки использовать матрицы, содержащие 5–10% ненулевых элементов с максимальным числом элементов 100.

Вариант схемы размещения матрицы определяется по формуле $((N + 3) \% 4) + 1$, где N — номер студента по списку в группе. Вариант преобразования определяется по формуле $((N – 1) \% 11) + 1$. Вариант физического представления (1 — отображение на массив, 2 — отображение на динамические структуры) определяются по формуле $([1.5\times((3+M) \% 9)]+N) \% 2+1$, где M — номер группы. В случае использования динамических структур индексы заменяются соответствующими ссылками.

Варианты схемы размещения матрицы: все матрицы $m \times n$ хранятся *по строкам*, в порядке возрастания индексов ненулевых элементов.

1. Цепочка ненулевых элементов в векторе *A* со строчным индексированием (индексы в массиве *M* равны 0, если соответствующая строка матрицы содержит только нули)

M:	Индекс начала 1-ой строки в массиве A	Индекс начала 2-й строки	...	Индекс начала N-ой Строки

A:	Номер столбца	Значение	Индекс следующего ненулевого элемента этой строки (или 0)	Номер столбца	Значение	Индекс следующего ненулевого элемента этой строки (или 0)	...

Индекс, равный нулю, означает отсутствие ненулевых элементов в строке (или в ее остатке).
Если матрицы не изменяются программой, возможна экономия памяти за счет отказа от хранения в массиве *A* индексов следующего элемента столбца (когда элементы идут подряд). Вставка и удаление при этом способе возможны, но чересчур д*о*роги: число перестановок элементов составит $O(N)$ вместо $O(1)$.

2. Один вектор:

Ненулевому элементу соответствуют две ячейки: первая содержит номер столбца, вторая содержит значение элемента. Нуль в первой ячейке означает конец строки, а вторая ячейка содержит в этом случае номер следующей хранимой строки. Нули в обеих ячейках являются признаком конца перечня ненулевых элементов разреженной матрицы.

0	Номер строки	Номер столбца	Значение	Номер столбца	Значение	...

...

0	Номер строки	Номер столбца	Значение	...	0	0

3. Три вектора:

CIP:	Индекс начала 1-ой строки в массивах PI и YE	Индекс начала 2-ой Строки	...	Индекс начала N-ой Строки

PI:	Номер столбца	Номер столбца	Номер столбца	...	Номер Столбца	0
	↓	↓	↓		↓	
YE:	Значение	Значение	Значение	...	Значение	

4. Два вектора:

LB:	λij	λij	λij	...	λij	-1
	↓	↓	↓		↓	
YE:	Значение	Значение	Значение	...	Значение	

где $\lambda_{ij} = (i - 1) \times n + j - 1$

По согласованию с преподавателем возможна модификация схемы хранения, например, хранение не исходной, а транспонированной матрицы.

Дополнительное задание: реализовать функциональную спецификацию АТД Матрица.

Варианты преобразований:

1. Определить максимальный по модулю элемент матрицы и разделить на него все элементы строки, в которой он находится. Если таких элементов несколько, обработать каждую строку, содержащую такой элемент.
2. Определить максимальный по модулю элемент матрицы и разделить на него все элементы столбца, в котором он находится. Если таких элементов несколько, обработать предпоследний столбец, содержащий такой элемент.
3. Найти элемент матрицы, ближайший к заданному значению. Разделить на него элементы строки и столбца, на пересечении которых он расположен. Если таких элементов несколько, обработать все.
4. Умножить разреженную матрицу на вектор-столбец и вычислить количество ненулевых элементов результата.
5. Умножить вектор-строку на разреженную матрицу и вычислить количество ненулевых элементов результата.
6. Вычислить сумму двух разреженных матриц. Проверить, не является ли полученная матрица симметричной.
7. Найти строку, содержащую наибольшее количество ненулевых элементов, и напечатать ее номер и сумму элементов этой строки. Если таких строк несколько, обработать все.
8. Вычислить произведение двух разреженных матриц. Проверить, не является ли полученная матрица диагональной.
9. Найти столбец, содержащий наибольшее количество ненулевых элементов, и напечатать его номер и произведение элементов этого столбца. Если таких столбцов несколько обработать предпоследний.
10. Вычислить матрочлен — многочлен первой степени от разреженной матрицы: ($a \cdot M + b \cdot E$), где E — единичная матрица, a и b — числовые константы.
11. Транспонировать разреженную матрицу относительно побочной диагонали. Выяснить, является ли полученная матрица кососимметрической.

Литература к заданию VII и к лабораторной работе № 21

1. *Никулин С.П. Представление и обработка данных с регулярной структурой // Под. ред. Зайцева В.Е. – М.: МАИ, 1997.*
2. *Берзтисс А.Т. Структуры данных. –М.: Мир, 1974.*
3. *Тьюарсон Р. Разреженные матрицы. –М.: Мир, 1977.*
4. *Писсанецки С. Технология разреженных матриц. –М.: Мир, 1980.*
5. *Карасев С.Б., Кошелева Т.Я., Чернышов Л.Н. Машинные алгоритмы обработки информации. –М.: Изд-во МАИ, 1987.*
6. *Кнут Д. Искусство программирования для ЭВМ. Т 1. Основные алгоритмы. –М.: Мир, 1976.*
7. *Кристиан К. Введение в операционную систему UNIX. –М.: Финансы и статистика, 1985.*
8. *Беляков И.Н., Рабовер Ю.И., Фридман А.Л. Мобильная операционная система: Справочник. –М.: Радио и связь, 1991.*
9. *Баурн С. Операционная система UNIX. –М.: Мир, 1986.*
10. *Тихомиров В.П., Давидов М.И. Операционная система ДЕМОС: инструментальные средства программирования. –М.: Статистика, 1988. (**C Shell, make**, lex, yacc).*
11. *Тейнсли Д. Linux и UNIX: программирование в shell. Руководство разработчика: Пер. с англ. –К.: Издательская группа BHV, 2001. –464 с.*
12. *Вирт Н. От Модулы к Оберону // Системная информатика. Вып.1. –Новосибирск: Наука, 1991, с.63–75.*

13. *Вирт Н. От разработки языка программирования к созданию компьютера. // Лекции лауреатов премии Тьюринга. – М.: Мир, 1993, с.210–223*
14. *FTP-архив Интернет МТИ: ftp://prep.ai.mit.edu/pub/gnu*
15. *Айлифф Дж. Принципы построения базовой машины. -М.: Мир, 1974.*
16. *Еремин А.Ю., Марьяшкин Н.Я. Пакет программ SPARSE для решения систем линейных алгебраических уравнений с разреженными матрицами. -М.: ВЦ АН СССР, 1978.*
17. *Еремин А.Ю., Марьяшкин Н.Я. Пакет программ SOLVER системы нелинейных функциональных и обыкновенных дифференциальных уравнений с разреженными якобиевыми матрицами. -М.: ВЦ АН СССР, 1980.*
18. *Белаш В.О., Марьяшкин Н.Я. Пакет программ СЛАУРМ. Решение систем с разреженными матрицами. -М.: ВЦ АН СССР, 1989.*

Вопросы для самостоятельного изучения к заданию VIII и к промежуточному зачёту VIII

ISO/IEC 9899:1999 Programming languages — C [C99]

1. Краткая историческая справка о предшественниках C: A, B, BCPL.
2. Стандарты языка C: C89 и C99. Новые возможности. Совместимость.
3. Дальнейшее развитие языка C: C++. Новые возможности. Совместимость.
4. Дополнительные типы данных C99: комплексный тип, целые типы. Булевский тип, заголовок <stdbool.h>.
5. Константные выражения. Недесятичная запись целых констант. Заголовочные файл <limits.h>, <stdint.h>.
6. Операторы сдвига << и >>. Операторы @=. Операторы *, & и ->. Оператор sizeof.
7. Строковый тип. Библиотека <string.h>.
8. Математическая библиотека <math.h>.
9. Функции в качестве типов и аргументов. Оператор typedef.
10. Структуры. Инициализация структур в C89 и C99.
11. Работа с файлами: библиотека <stdio.h>. Тип FILE, функции fopen, fclose, fread, fwrite, fscanf, fprintf, fseek. Стандартные файлы.
12. Выбор стандарта C. Опция –std.
13. Стандарт языка об особенностях вычисления логических выражений. Примеры.

СПИСКОВЫЕ СТРУКТУРЫ В ЯЗЫКАХ ПРОГРАММИРОВАНИЯ.

1. Моделирование списков на линейной памяти с прямым и последовательным доступом. Итераторы.
2. Списки в динамической памяти Си.
3. Библиотечные средства обработки списков в C++.
4. *Библиотечные средства обработки списков в Java.
5. *Представление списков деревьями (Пролог).
6. *Понятие о существенно списковых языках программирования (Лисп).
7. *Встроенные списки языка Python. Особенности индексации. Срезы. Встроенные операции над списками.

Задание VIII. Линейные списки

Составить и отладить программу на языке Си для обработки линейного списка заданной организации с отображением списка на динамические структуры (группы 1, 2, 3, 8, 9, 11, 13) или на массив (только с индексным доступом, без применения ссылок и указателей, для групп 4, 5, 6, 7, 10, 12). Навигацию по списку следует реализовать с применением итераторов. Предусмотреть выполнение одного нестандартного и четырех стандартных действий:

1. Печать списка.
2. Вставка нового элемента в список.
3. Удаление элемента из списка.
4. Подсчет длины списка.

ТИП ЭЛЕМЕНТА СПИСКА: (определяется как *номер_группы* % 8 + 1):

1. Целый.
2. Вещественный.
3. Перечислимый.
4. Строковый.
5. Литерный.
6. Комплексный.
7. Беззнаковое целое.
8. Булевский.
9. Двоично-кодированное десятичное (BCD).
10. *Множество.
11. *Ссылочный.
12. *Процедурный.

ВИД СПИСКА (определяется как (*N* div 2) % 6 + 1):

1. кольцевой однонаправленный;
2. линейный однонаправленный;
3. линейный однонаправленный с барьерным элементом;
4. кольцевой двунаправленный;
5. линейный двунаправленный;
6. линейный двунаправленный с барьерным элементом;

НЕСТАНДАРТНОЕ ДЕЙСТВИЕ (определяется как *N* % 15 + 1):

1. удалить из середины списка *k* элементов;
2. очистить список, если в нём есть элемент, равный заданному значению;
3. удалить из списка все элементы, предшествующие и последующие заданному значению;
4. обменять местами ($k - 1$) -й и ($k + 1$) -й элементы списка (*k* задается в качестве параметра);
5. обменять местами 2-й и предпоследний элементы списка;
6. удалить каждый *k*-ый элемент списка;
7. удалить элементы списка со значениями, находящимися в заданном диапазоне;
8. дополнить список копиями заданного значения до указанной длины *k*. Если в списке уже имеется не менее *k* элементов, то не менять его;
9. исключить из списка последние *k* элементов. Если в списке менее *k* элементов, то не менять его;
10. добавить *k* экземпляров последнего элемента в начало списка;
11. переставить элементы списка в обратном порядке;
12. проверить упорядоченность элементов списка;
13. выполнить циклический сдвиг элементов списка на один элемент вперед;
14. выполнить попарный обмен значениями элементов списка;
15. переставить первую и вторую половины списка.

При описании структур или алгоритмов задания VIII желательно использовать графическую иллюстрацию и/или нотацию одного из языков со встроенными списковыми структурами (LISP, Prolog).

Литература к заданию VIII

1. *Берзтисс А.Т. Структуры данных. —М.: Мир, 1974.*
2. *Разумов О.С. Организация данных в вычислительных системах. -М.: Статистика, 1978.*
3. *Донован Дж. Системное программирование. -М.: Мир, 1975.*
4. *Вирт Н. От Модулы к Оберону // Системная информатика. Вып.1 - Новосибирск: Наука, 1991, с.63–75.*
5. *Вирт Н. От разработки языка программирования к созданию компьютера. // Лекции лауреатов премии Тьюринга. – М.: Мир, 1993, с.210–223.*
6. *Кристиан К. Введение в операционную систему* UNIX. - *М.: Финансы и статистика, 1985.*
7. *Дейтел Г. Введение в операционные системы. Т.2- М.: Мир, 1987.*
8. *Цикритзис Д., Бернстайн Ф. Операционные системы. – М.: Мир, 1977.*
9. *Инструментальные средства разработки программ в современных операционных системах // Учебное пособие под ред. С.М.Юдина. - М.: Изд-во МАИ, 1990, с. 38-53, 78-85 (интерпретатор команд cshell).*
10. *Мейер Б., Бодуэн К. Методы программирования: в 2-х томах. –М.: Мир, 1982.*
11. *Луговая И.З., Чернышов Л.Н., Юдин С.М. Динамические структуры данных языка Паскаль. –М.: МАИ, 1988.*
12. *Баурн С. Операционная система UNIX. –М.: Мир, 1986.*
13. *Топхем Д., Хай Ван Чыонг. Юникс и Ксеникс. Пер. с англ. –М.: Мир, 1988.*
14. *Хювёнен Э., Сеппянен Й. Мир Лиспа. Введение в язык Лисп и функциональное программирование. – М.: Мир, 1990.*
15. *Лавров С.С., Силагадзе Г.С. Автоматическая обработка данных. Язык Лисп и его реализация. – М.: Наука, 1978.*
16. *Тихомиров В.П., Давидов М.И. Операционная система ДЕМОС. Инструментальные средства программирования. – М.: Финансы и статистика, 1988. с. 11-76.*
17. *Банахан М., Раттер Э. Введение в операционную систему UNIX. – М.: Радио и связь, 1986. с. 46-63.*

Вопросы для изучения к заданию IX

Сортировка и поиск

1. Таблицы. Статические и динамические таблицы. Задача поиска в таблицах.
2. Особенности поиска в упорядоченных и неупорядоченных таблицах.
3. Метод двоичного поиска.
4. Таблицы с прямым доступом.
5. Понятие о внутреннем и внешнем упорядочении. Терминология сортировок.
6. Обменные (пузырьковая, шейкер, …) сортировки.
7. Методы простой и двоичной вставки.
8. Методы линейного выбора с обменом и с подсчетом.
9. Турнирная сортировка.

10. Пирамидальная сортировка с просеиванием.
11. Быстрая сортировка Хоара: рекурсивный и нерекурсивный варианты.
12. Метод Шелла.
13. Сортировка простым двухпоточным слиянием.
14. Упорядочение с помощью дерева поиска.
15. Гладкая сортировка.

Системы программирования. Модульное программирование на Си

1. Модули в стандарте С. Директива #include. Стражи включения.
2. Экспорт идентификаторов.
3. Описание встроенных функций.
4. Разделение интерфейса и реализации.
5. Указатели. Бестиповый указатель. Указатель на функцию.
6. Объектные модули. Редактирование связей. Использование библиотек. Опция –l.
7. Реализация многоплатформенности в семействе GNU (GNU C).
8. Модульное многоязычие. Директива extern.
9. Автоматизация процесса компиляции и сборки модульных программ. Утилита **make**.
10. Особенности реализации СП GNU C для платформы MS Windows.

Задание IX. Сортировка и поиск

Составить программу на языке Си с использованием процедур и функций для сортировки таблицы заданным методом и двоичного поиска по ключу в таблице.

Программа должна **вводить** значения элементов неупорядоченной таблицы и проверять работу процедуры сортировки в трех случаях: (1) элементы таблицы с самого начала упорядочены; (2) элементы таблицы расставлены в обратном порядке; (3) элементы таблицы не упорядочены. В последнем случае можно использовать встроенные процедуры генерации псевдослучайных чисел.

Для каждого вызова процедуры сортировки необходимо печатать исходное состояние таблицы и результаты сортировки. После выполнения сортировки программа должна вводить ключи и для каждого из них выполнять поиск в упорядоченной таблице с помощью процедуры двоичного поиска и печатать найденные элементы, если они присутствуют в таблице.

В процессе отладки и тестирования рекомендуется использовать команды обработки текстовых файлов ОС UNIX и переадресацию ввода-вывода. Тестовые данные необходимо заранее поместить в текстовые файлы.

В качестве текста для записей таблицы взять фрагмент стихотворения (группы 2-5), прозы (группы 9-13) или изображение ASCII-графики (группы 1, 6-8). Каждый элемент таблицы, содержащий ключ и текст записи, распечатывать в отдельной строке.

Вариант задания определяется двумя числами: (1) - номер метода сортировки = ((N - 1) % 15) + 1, (2) - номер структуры таблицы = ((N + 5) % 9) + 1, где N - номер студента по списку в группе.

Метод сортировки (в терминах Н.Вирта [4,5] и Д. Кнута [2]):

1. Линейный выбор с обменом.
2. Линейный выбор с подсчетом.
3. Метод пузырька
4. Шейкер-сортировка.
5. Метод простой вставки.
6. Метод двоичной вставки.
7. Метод Шелла.
8. Турнирная сортировка.
9. Пирамидальная сортировка с просеиванием.
10. Простое двухпоточное слияние.
11. Быстрая сортировка Хоара (рекурсивный вариант).
12. Быстрая сортировка Хоара (нерекурсивный вариант).
13. Четно-нечетная сортировка (парный обмен, основанный на методе пузырька) **[1].**
14. Прямое слияние **[1].**
15. Естественное слияние **[1].**

16* Гладкая сортировка.

(Дополнительный вариант (*) для переводников и нарушителей учебно-производственной дисциплины).

Структура таблицы:

№	тип ключа	длина ключа байтах	хранение данных и ключей	минимальное число элемент таблицы
1	целый	8	вместе	11
2	целый	4	отдельно	12
3	строковый	5	отдельно	13
4	строковый	6	вместе	14
5	вещественный	16	вместе	15
6	вешественный	4	отдельно	16
7	комбинированный (целое + литера)	9	отдельно	17
8	комбинированный (строка + целое)	32	вместе	18
9	комплексный	16	вместе	19
10	комплексный	8	отдельно	20
11	кватернион	32	вместе	21
12	кватернион	32	отдельно	22

Литература к заданию IX

1. *Лорин Г. Сортировка и системы сортировки. –М.: Наука, 1983.*
2. *Кнут Д. Искусство программирования для ЭВМ. Т 3. Сортировка и поиск. –М.: Мир, 1976.*
3. *Карасев С.Б., Кошелева Т.Я., Чернышов Л.Н. Машинные алгоритмы обработки информации. –М.: Изд-во МАИ, 1987.*
4. *Разумов О.С. Организация данных в вычислительных системах. –М.: Статистика, 1978. - 184 с.*
5. *Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. –М.: МЦНМО, 2000. – 960 с., ил.*
6. *Шень А. Программирование. Теоремы и задачи. –М.: МЦНМО, 1995.*
7. *Гасфилд Д. Строки, деревья и последовательности в алгоритмах: Информатика и вычислительная биология/ Пер. с англ. И.В. Романовского. –СПб: Невский Диалект, БХВ-Петербург, 2003. –654 с., ил.*

Темы рефератов курсового проекта по информатике во II семестре 2020/21 учебного года (при его наличии)

1. Алгоритмы и структуры данных на языке Си: достоинства и недостатки.
2. Тестирование и отладка (применительно к СП Си, ОС UNIX).
3. Интерпретируемые командные языки ОС UNIX (история, обзор, сравнение, программирование).
4. Модульное программирование и Паскаль (Modula-2, TurboPascal, GNU Pascal, Modula-3, Oberon, Ada).
5. Языковая среда Си. Библиотека. Сравнение с C++ и Java.
6. Обработка текстов в ОС UNIX.
7. Рекурсивные методы в программировании.
8. *Объектно-ориентированное программирование (на примере одного из языков Object Pascal, C++, CLOS, Smalltalk, Eiffel, Java, C#, Python).
9. Сравнение языков программирования Паскаль и Си.
10. Разреженные матрицы.
11. TeX: полный по Тьюрингу язык программирования блочной структуры.
12. Файлы и базы данных (Паскаль, UNIX, SQL, ...).
13. *Логическое программирование.
14. *Функциональное программирование. LISP, F#, Haskell или Ruby
15. *Продукционное программирование. РЕФАЛ.

Темы со звездочкой требуют самостоятельной работы и даются по согласованию с преподавателем.

Темы лабораторных занятий (68-102 часа)

1. Файловые утилиты ОС UNIX.
2. Создание и обработка внешних текстовых и нетекстовых файлов в среде СП Си.
3. Программирование на интерпретируемых командных языках.
4. Стек и дек (отображение на динамические структуры).
5. Разреженные матрицы.
6. Линейный список на массиве. Кольцевые и двунаправленные списки. Сборка мусора. Список свободных элементов.
7. Линейный список на динамических структурах. Итераторы.
8. Дерево и двоичное дерево. Инициализация, добавление узла, удаление узла, визуализация. Обход двоичного дерева (прямой, обратный, концевой). Обход дерева общего вида в глубину, в ширину. Прошитые деревья.
9. Обход дерева с использованием стека. Примеры: рекурсивное вычисление высоты дерева, вычисление высоты дерева без рекурсии с использованием стека или прошивки.

10. Представление выражения в виде дерева. Польская инверсная запись. Обработка и вычисление выражений (рекурсия или стек).
11. Поиск путей в графе (в глубину с рекурсией, в глубину без рекурсии, в ширину).
12. Поиск по образцу.
13. Абстрактные типы данных. Противопоставление разреженным матрицам. АТД и процедурное программирование.
14. Реализация стека и дека на массиве. Пример рекурсивной обработки стека (вставка и удаление элемента с заданным номером). Визуализация Ханойских башен.
15. Стек: модуль определений, модуль реализации и программный модуль с меню.
16. Сортировка таблиц (внутренняя).
17. Сортировка последовательностей (внешняя).
18. Хэш-таблицы.
20. Обработка текстовых файлов в ОС UNIX.
21. Программирование на командном языке ОС UNIX.
22. Математическое издательство в среде **TeX.**
23. Динамические структуры данных. Обработка деревьев.
24. Рекурсивные методы и структуры данных. Представление и обработка выражений.
25. Абстрактные типы данных. Сортировка последовательностей.
26. Автоматизация процесса сборки программ модульной структуры с использованием утилиты **make**.

Дополнительно (по согласованию с преподавателем): выполнение одного из заданий практикума в альтернативных средах программирования.

27.* Объектно-ориентированное программирование в фон-Неймановских языках (C++, Java, C# и др.).
28.* Программирование в абсолютно объектных средах (CLOS, SmallTalk).
29.* Программирование на Прологе, Лиспе или в системе AFP Дж. Бэкуса.

Задания к лабораторным работам представлены отдельными документами.

Темы консультаций по курсу (контролируемая самостоятельная работа не менее 34 часов)

1. Подготовка к лабораторной работе 20. Команды обработки текстовых файлов в ОС UNIX.
2. Задание VI курсового проекта. Хранение и выборка данных в бинарных файлах.
3. Входной контроль знаний по заданию VI.
4. Подготовка к лабораторной работе 21. Программирование на ИКЯ ОС UNIX.
5. Подготовка к лабораторной работе 22. Издательская система TeX.
6. Задание VII курсового проекта. Разреженные матрицы.
7. Входной контроль знаний по заданию VII.
8. Задание VIII курсового проекта. Линейные списки.
9. Входной контроль знаний по заданию VIII.
10. Подготовка к лабораторной работе 23.
11. Подготовка к лабораторной работе 24.
12. Подготовка к лабораторной работе 25.
13. Задание IX курсового проекта. Методы сортировки и поиска.
14. Входной контроль знаний по заданию IX.
15. Подготовка к лабораторной работе 26.
16. Проверка отчётов по заданиям практикума, тестирование и инспекция программ.
17. Консультация по экзаменационным задачам.

Программа экзамена/рейтингового зачёта по курсу «Языки и методы программирования» Институт № 8, 1 курс, II семестр 2021/22 уч. года (68 часов лекций)

Утверждаю:

Зав. кафедрой 806,
член-корр. РАН Пирумов У.Г.

1. Уровни описания структур данных.
2. Статические и динамические объекты программ.
3. Ссылочный тип данных.
4. Файл. Функциональная спецификация.
5. Файл. Логическое описание. Физическое представление.
6. Вектор. Функциональная спецификация. Логическое описание и физическое представление.
7. Очередь. Функциональная спецификация.
8. Очередь. Логическое описание и физическое представление (файл).

9. Очередь. Логическое описание и физическое представление (массив).
10. Очередь. Логическое описание и физическое представление (динамические объекты).
11. Стек. Функциональная спецификация.
12. Стек. Логическое описание.
13. Стек. Физическое представление (массив).
14. Стек. Физическое представление (динамические объекты).
15. Линейный список. Функциональная спецификация.
16. Линейный список. Логическое описание.
17. Линейный список. Физическое представление. Итераторы.
18. Линейный список. Физическое представление (массив).
19. Линейный список. Физическое представление (динамические объекты).
20. Списки общего вида. Представление и обработка графов.
21. Рекурсивные структуры данных.
22. Деревья. Двоичные деревья.
23. Двоичное дерево. Функциональная спецификация.
24. Двоичное дерево. Логическое описание. Построение и визуализация.
25. Двоичное дерево. Физическое представление. Прошивка.
26. Алгоритмы обхода деревьев.
27. Особенности представления и обработки деревьев общего вида.
28. Деревья выражений.
29. Деревья поиска.
30. Сбалансированные деревья поиска.
31. Задача поиска. Простые методы поиска в последовательностях и таблицах.
32. Алгоритм Кнута-Морриса-Пратта.
33. Алгоритм Бойера-Мура.
34. Алгоритм Рабина-Карпа.
35. Таблицы с прямым доступом.
36. Задача сортировки.
37. Сортировка вставкой.
38. Сортировка выборкой.
39. Обменные сортировки.
40. Сортировка Шелла.
41. Турнирная сортировка.
42. Пирамидальная сортировка.
43. Сортировка Хоора.
44. Гладкая сортировка.
45. Сортировка слиянием.
46. Сортировка естественным слиянием.
47. Сравнение методов сортировки.
48. Процедурное программирование.
49. Модульное программирование. Реализация на языке Си.
50. Абстракции в языках программирования.
51. Абстрактные типы данных. Пример модуля АТД *ОЧЕРЕДЬ*.
52. Экспорт и импорт объектов. Инкапсулированные АТД.
53. Типизация языка программирования. Контроль типов.
54. Средства ослабления типового контроля. Преобразование и передача типов.
55. Полиморфизм операций, отношений, процедур, функций и модулей.
56. Адресный тип.
57. Реализация полиморфизма с помощью адресного типа.
58. Процедурный тип данных.
59. Реализация полиморфизма с помощью процедурного типа.
60. Наследование.
61. Реализация полиморфизма с помощью наследования.
62. Парадигма функционального программирования.
63. Парадигма логического программирования.

В качестве экзаменационных задач предлагаются задачи на написание программ на C, а также на Shell, или, в отдельных случаях, на TeX по всем концепциям, алгоритмам и структурам данных теоретического курса и практикума.

Программа может быть уточнена не позднее последней лекции (или предпоследней для варианта курса с рейтинговым зачётом).

Программу составил лектор курса Зайцев В.Е.

Лист переутверждения от 31 августа 2021 г.

Зав. кафедрой 806 Крылов С.С.