АВС, ИДЗ 3, Самойлов Павел Павлович.

Вариант 21:

Задача об инвентаризации по рядам. После нового года в библиотеке университета обнаружилась пропажа каталога. После поиска и наказания виноватых, ректор дал указание восстановить каталог силами студентов. Фонд библиотека представляет собой прямоугольное помещение, в котором находится M рядов по N шкафов по K книг в каждом шкафу. Требуется создать многопоточное приложение, составляющее каталог. При решении задачи использовать метод «портфель задач», причем в качестве отдельной задачи задается составление каталога одним студентом для одного ряда. Примечание. Каталог — это список книг, упорядоченный по их инвентарному номеру или по алфавиту (на выбор разработчика). Каждая строка каталога содержит идентифицирующее значение (номер или название), номер ряда, номер шкафа, номер книги в шкафу.

Модель параллельных вычислений:

В данной задаче я буду использовать парадигму портфеля задач. У нас такой есть (библиотека), затем каждый поток берет себе задачу и выполняет действие (сортировка).

Отсчет:

4 балла:

1) Приведено условие задачи. 2) Описана модель параллельных вычислений, используемая при разработке многопоточной программы. 3) Описаны входные данные программы, включающие вариативные диапазоны, возможные при многократных запусках. и скомпонована без использования опций отладки 4) Реализовано консольное приложение, решающее поставленную задачу с использованием одного варианта синхропримитивов. 5) Ввод данных в приложение реализован с консоли.

5 баллов:

1) В программу добавлены комментарии, поясняющие выполняемые действия и описание используемых переменных. 2) В отчете должен быть приведен сценарий, описывающий одновре- менное поведение представленных в условии задания сущностей в терминах предметной области. То есть, описано поведение объектов разрабатываемой программы как взаимодействующих субъектов, а не то, как это будет реализовано в программе.

6-8 баллов:

1) В программу добавлены ввод данных из файла и вывод результатов в файл. 2) В программу добавлена генерация случайных данных в допустимых диапазонах.

Подробное описание проведенной работы:

Реализуем программу на C++(Сразу код на 8 баллов) и закомментируем все важные моменты.

```
#include <algorithm>
#include <fstream>
#include <iostream>
#include <pthread.h>
```

```
// Мьютекс и текущая позиция в бибилиотеке
pthread_mutex_t mutex;
int curN = 0;

// Текущий поток
int cnt = 0;

// Библиотека
const int n = 10;
const int m = 20;
int library[n][m];

std::ifstream fin; // Поток для чтения.
std::ofstream fout; // Поток для вывода.

// Метод сортировки книг по каталогам
void *sortBooks(void *param)
{
  // Блок мьютекса
  pthread_mutex_lock(&mutex);
  // Соритровка книжек
  std::sort(std::begin(library[curN]), std::end(library[curN]));
  std::cout << "Shelve num " << curN << " was sorted. \n";
  fout << "Shelve num " << curN << " was sorted. \n";
  for (int i = 0; i < m; ++i) {
    std::cout << library[curN][i] << " ";
  }
  std::cout << '\n';
  curN++;
  pthread_mutex_unlock(&mutex);// unlock thread
}

int main(int argc, char **argv) {
  // Рандомное число, argc == 2 рандом
  int randNum = 1;
  if (argc == 2) {
    char *arg = argv[1];
    int randSeed = atoi(arg);
    srand(randSeed);
    randNum = rand() % 30;
    // Считываение из файла
  } else if (argc == 3) {
    fin.open(argv[1]);
    fout.open(argv[2]);

    if (!fin.is_open() || !fout.is_open()) {
        std::cout << "Проблемы с открытием файла.";
        return 0;
    }

    fin >> randNum;
```

```
  } else {
    std::cout << "Enter num for books: " << '\n';
    std::cin >> randNum;

    fout.open("output.txt");
  }

  // Номер каталога
  int libRand = 1000;
  // Заполнение библиотеке
  for (int i = 0; i < n; ++i) {
    for (int j = 0; j < m; ++j) {
      library[i][j] = libRand;
      libRand -= randNum;
    }
  }

  // Поток студентов
  pthread_t students[n];

  // Инициализация мьютекса
  pthread_mutex_init(&mutex, 0);

  // Портфель задач. Каждый студент разбирает своё задание ( то есть сортировку )
  for (int i = 0; i < n; ++i) {
    pthread_create(&students[i], 0, sortBooks, (void *) cnt);
    std::cout << "Student  " << i << " started" << '\n';
    fout << "Student  " << i << " started" << '\n';
  }

  // Выполнение параллелизации
  for (int i = 0; i < n; ++i) {
    pthread_join(students[i], 0);
    std::cout << "Student " << i << "  finished" << '\n';
    fout << "Student " << i << "  finished" << '\n';
  }

  pthread_mutex_destroy(&mutex);
  return 0;
}
```

Выходные данные:

```
Enter num for books:
5
Student  Shelve num 0 started
0 was sorted.
905 910 915 920 925 930 935 940 945 950 955 960 965 970 975 980 985 990 995 1000
Student  1 started
Shelve num 1 was sorted.
805 810 815 820 825 830 835 840 845 850 855 860 865 870Student  2 started
 875 880 885 890 895 900
```

```
Student  3 started
Shelve num 2 was sorted.
705 710 715 720 725 730 735 740 745 750 755 760 765 770 775 780 785 790 795 800
Student  4 started
Shelve num 3 was sorted.
605 610 615 620 625 630 635 640 645 650 655 Student  5 started
660 665 670 675 680 685 690 695 700
Student  6 started
Shelve num 4 was sorted.
505 510 515 520 525 530 535 540 545 550 555 560 565 570 575 580 585 590 595 600
Student  7 started
Shelve num 5 was sorted.
405 410 415 420 425 430 435 440 445 450 455 460 465Student  8 started
 470Student  9 started
 475 480 485 490 495 500
Student 0  finished
Shelve num 6 was sorted.
305 310 315 320 325 330 335 340 345 350 355 360 365 370 375 380 385 390 395 400
Student 1  finished
Shelve num 7 was sorted.
205 210 215 220 225 230 235 240 245 250 255 260 265 270 275 280 285 290 295 300
Student 2  finished
Shelve num 8 was sorted.
105 110 115 120 125 130 135 140 145 150 155 160 165 170 175 180 185 190 195 200
Student 3  finished
Shelve num 9 was sorted.
5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100
Student 4  finished
Student 5  finished
Student 6  finished
Student 7  finished
Student 8  finished
Student 9  finished
```

Как видно из выходных данных, программа успешно работает, в соответсвии с выбранной моделью решения.

(Также в программе есть возможность рандомной генерации данных для этого нужно ввести 'seed'), а также передача и вывод данных в файл, для этого нужно передать input.txt, output.txt.

Пример запуска программы для данных опций:

```
./main input.txt output.txt
./main 5
```

(При неправильном вводе входного/выходного файла программа выдаст ошибку в консоль.)
(Результаты тестирования в папке Tests)