Министерство науки и высшего образования Российской Федерации

Новосибирский государственный технический университет

Управление ресурсами в вычислительных системах

Лабораторная работа №7

Факультет: прикладной математики и информатики

Группа: ПМ-63

Студенты: Кожекин М.В.

Утюганов Д.С.

Преподаватели: Стасышин В. М.

Хайленко Е. А.

Новосибирск

2019

1. Условие (Вариант №5)

Программа моделирует работу примитивной СУБД, хранящей единственную таблицу в оперативной памяти. Выполняя некоторые циклы работ, K порожденных процессов посредством очереди сообщений передают родительскому процессу номер строки и её содержимое, на которое нужно изменить хранящиеся в ней данные. Родительский процесс выполняет указанную операцию и возвращает старое содержимое изменённой строки.

1. Анализ задачи

Порождаем очередь сообщений.

Создаём таблицу.

Порождаем К процессов-потомков, каждый из которых подготавливает изменение строки таблицы и отправляет его через очередь сообщений процессу-предку.

Процесс-предок производит изменения исходной таблицы согласно командам из очереди сообщений и выводит полученную таблицу на экран.

Уничтожаем очередь сообщений.

1. Используемые программные средства

**Функции:**

int fork() – порождение процесса-потомка

int msgget msgget(key\_t key, int flag) – создание очереди

int msgsnd (int msgqid, void \*msg, size\_t size, int flag) – отправка сообщения

int msgrcv (int msgqid, void \*msg, size\_t size, long msg\_type, int flag) – приём сообщений

int msgctl (int msgqid, int command, struct msqid\_ds \*msg\_stat) – уничтожение очереди

int wait(int \*status) – ожидание завершения процесса-потомка

int getpid() – определение pid текущего процесса

int exit(int code) – прерывание процесса

int sprintf(char \*dest, const char \* format, …) – вывод данных в массив

int strncpy(char \*dest, char \*src, int length) – копирование строки **src** длиной **length** в **dest**

1. Спецификация

Программа находится в папке ~/lab7

Чтобы собрать программу нужно ввести “make lab7”

Чтобы запустить программу, нужно использовать команду “./lab7”

В результате работы программы, будет смоделирована работа примитивной СУБД.

1. Исходный код

**Файл lab7.c**

|  |
| --- |
| #include <signal.h>  #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include <sys/types.h>  #include <unistd.h>  #include <sys/ipc.h>  #include <sys/msg.h>  #include <fcntl.h>  #include <errno.h>  #define LENGTH 64  #define K 5  struct msg {      long mtype;      char mtext[LENGTH];  };  char table[K][LENGTH];  // Вывод таблицы на экран  void printTable()  {      printf("Таблица:\n");      for (size\_t i = 0; i < K; i++)          printf("%d:\t%s\n", i, table[i]);  }  // Замена строки по команде от процесса-потомка  void replaceTableLine(char \*s){      int i =0;      int number = 0;      int len = strlen(s);      while (isdigit(s[i]) && i<len)      {          number\*=10;          number+=s[i] - '0';          i++;      }      strncpy(table[number], s+i, len - i);  }  int main()  {      struct msg message;      // Создаём очередь сообщений      int msgQueueID;      key\_t key;      if (key = ftok("lab7", "a") == -1)      {          printf("ftok error\n");          exit(-1);      }      if ((msgQueueID = msgget(key, IPC\_CREAT | 0666)) == -1)      {          printf("Ошибка создания очереди\n");          exit(-1);      }      // Инициализируем таблицу начальными значениями      for (size\_t i = 0; i < K; i++)          sprintf(table[i], "Строка номер %d", i);      // Выводим исходную таблицу      printTable();      // Изменяем содержимое таблицы в процессах-потомках      for (size\_t i = 0; i < K; i++)      {          if (fork() == 0)          {              printf("Процесс %d(%d) изменяет строку %d\n", i, getpid(), i);              message.mtype = 1;              sprintf(message.mtext, "%d Новая строка номер %d", i, i);              // Отправляем через очередь сообщений              if (msgsnd(msgQueueID, &message, sizeof(message.mtext), IPC\_NOWAIT) < 0)              {                  printf("Send error\n");                  exit(-1);              }              exit(0);          }      }      // Ожидаем завершение K процессов-потомков      int status, wpid;      while ((wpid = wait(&status)) > 0);      for (size\_t i = 0; i < K; i++)      {          // Получем сообщение из очереди          msgrcv(msgQueueID, &message, sizeof(message.mtext), 1, IPC\_NOWAIT);          // Заменяем строку таблицы          replaceTableLine(message.mtext);          // Выводим модифицированную таблицу          printTable();      }      // Уничтожаем очередь сообщений      msgctl(msgQueueID, IPC\_RMID, NULL);      return 0;  } |

**Содержимое файла Makefile**

|  |
| --- |
| # Makefile for lab #7  lab7:  gcc lab7.c -std=gnu99 -o lab7  clean:  rm lab7  run:  rm lab7 lab7.c; nano lab7.c; gcc lab7.c -std=gnu99 -o lab7; ./lab7; |

1. Результат работы программы

Программа работает должным образом

|  |
| --- |
| -bash-4.2$ ./lab7  Таблица:  0: Строка номер 0  1: Строка номер 1  2: Строка номер 2  3: Строка номер 3  4: Строка номер 4  Процесс 0(53872) изменяет строку 0  Процесс 2(53874) изменяет строку 2  Процесс 1(53873) изменяет строку 1  Процесс 4(53876) изменяет строку 4  Процесс 3(53875) изменяет строку 3  Таблица:  0: Новая строка номер 0  1: Строка номер 1  2: Строка номер 2  3: Строка номер 3  4: Строка номер 4  Таблица:  0: Новая строка номер 0  1: Строка номер 1  2: Новая строка номер 2  3: Строка номер 3  4: Строка номер 4  Таблица:  0: Новая строка номер 0  1: Новая строка номер 1  2: Новая строка номер 2  3: Строка номер 3  4: Строка номер 4  Таблица:  0: Новая строка номер 0  1: Новая строка номер 1  2: Новая строка номер 2  3: Строка номер 3  4: Новая строка номер 4  Таблица:  0: Новая строка номер 0  1: Новая строка номер 1  2: Новая строка номер 2  3: Новая строка номер 3  4: Новая строка номер 4 |