Қазақстан Республикасының
Білім және ғылым
министрлігі

Министерство
образования и науки
Республики Казахстан

Д. Серікбаев атындағы
ШҚМТУ

ВКГТУ им. Д. Серикбаева

УТВЕРЖДАЮ

декан ФИТЭ
Г.Х.Мухамедиев
_____2012 г.

ЖҮЙЕЛІК БАҒДАРЛАМАЛАУ

зертханалық жұмыстарға арналған әдістемелік нұсқаулар

СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ

методические указания к лабораторным работам, СРС и СРСП

Специальность: 6В070400 «Вычислительная техника и программное обеспечение»
Форма обучения: очная

Өскемен
Усть-Каменогорск
2012

УДК 681.3 (06)

Никифоров В.Л. Системное программирование: Методические указания по дисциплине «Системное программирование» разработаны для более глубокого изучения языка C#. Специальность: 5В070400 «Вычислительная техника и программное обеспечение»
/ ВКГТУ.- Усть-Каменогорск, 2012.- 32 с.

Методические указания содержит теоретический материал, большое количество примеров решения задач, вариантов индивидуальных заданий для СРС и вопросов, подготовка которых необходима для защиты отчетов лабораторных работ на СРСП. Изложенный материал будут способствовать глубокому усвоению теоретического курса дисциплины «Системное программирование», формированию практических навыков по созданию программ на языке C# в консольном приложении среды Visual Studio.NET 2010.

Утверждено методической комиссией факультета информационных технологий и информатики.

Протокол № от 2012г.

СОДЕРЖАНИЕ

ТЕМА 1 СОЗДАНИЕ DLL БИБЛИОТЕКИ НА ЯЗЫКЕ C#

1.1 Цель лабораторной работы

Получить практические навыки по созданию и использованию динамически подключаемых библиотек (DLL) в языке C#.

1.2 Методические указания к лабораторной работе

Изучить материал 1 лекции дисциплины.

Изучить материал, рекомендуемый в силлабусе: стр. 275- 279 [2] – сокращенный вариант представлен:

«Любая библиотека — это сервер, предоставляющий свои ресурсы клиентам. Создадим клиентское приложение, выполняющее те же функции, что и приложение из раздела «Виртуальные методы», но с использованием библиотеки MonsterLib.dll. Для того чтобы компилятор мог ее обнаружить, необходимо после создания проекта (как обычно, это — консольное приложение) подключить ссылку на библиотеку с помощью команды Project ▸ Add Reference (добавить ссылку). Для поиска каталога, содержащего библиотеку, следует использовать кнопку Browse.

После подключения библиотеки можно пользоваться ее открытыми элементами таким же образом, как если бы они были описаны в том же модуле.»

1.3 Домашнее задание на лабораторную работу

Разработать DLL для работы с переменными типа string. В библиотеку включить методы, позволяющие выполнять сложение, сравнение и замену строчных символов на прописные символы. В программе предусмотреть печать результатов (или необходимых комментарий) после выполнения любого метода DLL. Значение переменных задавать в режиме диалога. В программе использовать меню.

1.4 Индивидуальные задания для СРС

Программы всех индивидуальных заданий должны иметь меню, в котором необходимо использовать включаемые в DLL методы.

1.4.1 Разработать DLL для работы с массивами целых чисел. В библиотеку включить метод поиска максимального элемента массива, метод поиска минимального элемента массива и метод обмена значениями максимального и минимального элементов массива. В программе предусмотреть создание и печать массива, обмен значениями максимального и минимального элементов и печать нового массива.

1.4.2 Разработать DLL для работы с целыми и вещественными числами. В библиотеку включить методы печати чисел в форматах E, F и C. В программе предусмотреть анализ на соответствие выводимого числа заданному формату и печать числа после выполнения любого метода DLL. Числа задавать в режиме диалога.
1.4.3 Разработать DLL для работы с переменными типа DateTime. В библиотеку включить методы, позволяющие по значению заданной переменной типа DateTime отдельно печатать значение года, месяца и дня недели. В программе предусмотреть печать результатов после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.
1.4.4 Разработать DLL для работы с переменными типа Char. В библиотеку включить методы, позволяющие по значению заданной переменной типа Char, определить является ли заданный символ цифрой, пробелом и управляющим символом. В программе предусмотреть печать результатов (или необходимых комментарий) после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.
1.4.5 Разработать DLL для работы с массивами целых чисел. В библиотеку включить метод сортировки только четных значений элементов массива, метод сортировки только нечетных значений элементов массива и метод сортировки всех значений массива. Все сортировки выполнять по убыванию. В программе предусмотреть создание и печать исходного массива, печать массива после любого вида сортировки.
1.4.6 Разработать DLL для работы с целыми и вещественными числами. В библиотеку включить методы печати чисел в форматах C, N и X. В программе предусмотреть анализ на соответствие выводимого числа заданному формату и печать числа после выполнения любого метода DLL. Числа задавать в режиме диалога.
1.4.7 Разработать DLL для работы с переменными типа DateTime. В библиотеку включить методы, позволяющие по значению заданной переменной типа DateTime отдельно печатать значение часа, минуты и секунды. В программе предусмотреть печать результатов после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.
1.4.8 Разработать DLL для работы с переменными типа Char. В библиотеку включить методы, позволяющие по значению заданной переменной типа Char, определить является ли заданный символ буквой, прописной буквой, латинской буквой. В программе предусмотреть печать результатов (или необходимых комментарий) после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.
1.4.9 Разработать DLL для работы с массивами целых чисел. В библиотеку включить метод сортировки только положительных значений элементов массива, метод сортировки только отрицательных значений элементов массива и метод сортировки всех значений массива. Все сортировки выполнять по убыванию. В программе предусмотреть создание и печать исходного массива, печать массива после любого вида сортировки.

1.4.10 Разработать DLL для исследования операций сравнения <=, >=, ==, != при работе с целыми и вещественными числами. В библиотеку включить методы для каждой операции сравнения. В программе предусмотреть печать результатов сравнения после выполнения любого метода DLL. Числа задавать в режиме диалога.
1.4.11 Разработать DLL для работы с переменными типа DateTime. В библиотеку включить методы, позволяющие по значению заданной переменной типа DateTime отдельно печатать секунды, миллисекунды и тики. В программе предусмотреть печать результатов после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.
1.4.12 Разработать DLL для работы с переменными типа Char. В библиотеку включить методы, позволяющие по значению заданной переменной типа Char, определить является ли заданный символ буквой, строчной буквой, кириллицей. В программе предусмотреть печать результатов (или необходимых комментарий) после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.
1.4.13 Разработать DLL для работы с массивами целых чисел. В библиотеку включить метод сдвига значений элементов массива на один разряд влево, метод сдвига значений элементов массива на один разряд вправо и метод обмена значениями всех элементов массива – 1 с N, 2 с N-1 и т.д. В программе предусмотреть создание и печать исходного массива, печать массива после выполнения любого метода DLL.
1.4.14 Разработать DLL для исследования логических операций &&, | | и ! при работе с двумя логическими переменными. В библиотеку включить методы для каждой логической операции, при этом необходимо перебирать все логические комбинации двух переменных. В программе предусмотреть печать результатов после выполнения любого метода DLL.
1.4.15 Разработать DLL для работы с переменными типа DateTime. В библиотеку включить методы, позволяющие по значению заданной переменной типа DateTime отдельно печатать название месяца, номер месяца в году и год. В программе предусмотреть печать результатов после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.
1.4.16 Разработать DLL для работы с переменными типа Char. В библиотеку включить методы, позволяющие по значению заданной переменной типа Char, определить является ли заданный символ знаком пунктуации, разделителем или пробелом. В программе предусмотреть печать результатов (или необходимых комментарий) после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.
1.4.17 Разработать DLL для работы с массивами целых чисел. В библиотеку включить метод поиска первого максимального элемента массива и перемещение его в начало массива, метод поиска второго максимального элемента массива и перемещения его в конец массива, поиск третьего элемента массива и перемещения его в середину массива. В программе

предусмотреть создание и печать массива, печать массива после выполнения любого метода DLL.
1.4.18 Разработать DLL для исследования операций /, % и * при работе с целыми и вещественными числами. В программе предусмотреть печать результатов после выполнения любого метода DLL. Числа задавать в режиме диалога.
1.4.19 Разработать DLL для работы с переменными типа DateTime. В библиотеку включить методы, позволяющие по значению заданной переменной типа DateTime отдельно печатать количество дней до «Нового года», номер дня в году и год, месяц, день. В программе предусмотреть печать результатов после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.
1.4.20 Разработать DLL для работы с переменными типа Char. В библиотеку включить методы, позволяющие по значению заданной переменной типа Char, определить является ли заданный символ математическим символом, числовым символом, Unicode-символом. В программе предусмотреть печать результатов (или необходимых комментарий) после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.

1.5 Контрольные вопросы для защиты отчета на СРСП

1.5.1 Что из компонентов компьютера можно отнести к физическим ресурсам компьютера? Примеры.
1.5.2 Как называются программы, работающие на компьютере под управлением ОС? Примеры.
1.5.3 Какие функции интерфейса программирования приложений в Windows в основном изучаются в дисциплине системного программирования?
1.5.4 Что является объектом в системе Windows?
1.5.5 Какие объекты интерфейса программирования приложений в Windows в основном изучаются в дисциплине системного программирования?
1.5.6 Где в системе Windows находятся функции Win32 API? Примеры.
1.5.7 Зачем нужны динамически подключаемые библиотеки?
1.5.8 Какой тип проекта имеют динамически подключаемые библиотеки?
1.5.9 Как DLL подключаются к проекту?
1.5.10 Понятие компьютера?

ТЕМА 2 ИСПОЛЬЗОВАНИЕ УКАЗАТЕЛЕЙ В ЯЗЫКЕ C#

2.1 Цель лабораторной работы

Получить практические навыки по объявлению указателей в консольных приложениях и их использованию в языке C#. Освоить операции с указателями при работе с массивами и другими структурами данных.

2.2 Методические указания к лабораторной работе

Изучить материал 2, 3 и 4 лекций дисциплины.

Изучить материал, рекомендуемый в силлабусе: стр. 347- 355 [2] – сокращенный вариант представлен:

«Небезопасный код

Одним из основных достоинств языка C# является его схема работы с памятью: автоматическое выделение памяти под объекты и автоматическая уборка мусора. При этом невозможно обратиться по несуществующему адресу памяти или выйти за границы массива, что делает программы более надежными и безопасными и исключает возможность появления целого класса ошибок, доставляющих массу неудобств при написании программ на других языках.

Однако в некоторых случаях возникает необходимость работать с адресами памяти непосредственно, например, при взаимодействии с операционной системой, написании драйверов или программ, время выполнения которых критично. Такую возможность предоставляет так называемый небезопасный (unsafe) код.

Небезопасным называется код, выполнение которого среда CLR не контролирует. Он работает напрямую с адресами областей памяти посредством указателей и должен быть явным образом помечен с помощью ключевого слова unsafe, которое определяет так называемый небезопасный контекст выполнения.

Ключевое слово unsafe может использоваться либо как спецификатор, либо как оператор. В первом случае его указывают наряду с другими спецификаторами при описании класса, делегата, структуры, метода, поля и т. д. — везде, где допустимы другие спецификаторы. Это определяет небезопасный контекст для описываемого элемента, например:

```
public unsafe struct Node
{
  public int   Value;
  public Node* Left;
  public Node* Right;
}
```

Вся структура Node помечается как небезопасная, что делает возможным использование в ней указателей Left и Right. Можно применить и другой вариант описания, в котором небезопасными объявляются только соответствующие поля структуры:

```
public struct Node
{
  public int Value;
  public unsafe Node* Left;
  public unsafe Node* Right;
}
```

Оператор unsafe имеет следующий синтаксис:

unsafe блок

Все операторы, входящие в блок, выполняются в небезопасном контексте.
ПРИМЕЧАНИЕ
Компиляция кода, содержащего небезопасные фрагменты, должна производиться с ключом /unsafe. Этот режим можно установить путем настройки среды Visual Studio (Project ▸ Properties ▸ Configuration Properties ▸ Build ▸ Allow Unsafe Code).»

2.3 Домашнее задание на лабораторную работу

Разместить в динамической памяти компьютера матрицу 5x5 случайных целых чисел в диапазоне минус 60 до 60. Напечатать ее. Предусмотреть оба варианта использования указателей для работы с массивами (один указатель на массив и массив указателей).

2.4 Индивидуальные задания для СРС

2.4.1 Очередь в столовую 100 человек представлена 4 категориями людей: студенты, преподаватели, школьники и сотрудники. Очередь формируется случайным образом с одинаковой вероятностью для каждой категории. Напечатать общую очередь. Разделить общую очередь на две очереди – для преподавателей и сотрудников, студентов и школьников. Напечатать обе очереди.
2.4.2 Создать стек 25 студентов, в каждый элемент которого включать номер студента и его оценку за экзамен. Оценки формировать случайным образом в диапазоне от 2 до 5. Напечатать стек. Разделить стек на два отдельных стека – «отличников» и «хорошистов», «троечников» и «двоечников». Напечатать оба стека.
2.4.3 Создать стек 20 случайных чисел в диапазоне от минус 0 до 100. Напечатать его. Поменять местами первый и последний элементы стека. Обмен выполнить с помощью указателей, а не «перезаписью» содержимого элементов. Напечатать новый стек.
2.4.4 Создать очередь 20 случайных чисел в диапазоне от минус 60 до 60. Напечатать ее. «Переписать» числа в новую очередь так, чтобы положительные числа разместились в начале очереди, а отрицательные числа – в конце очереди. Напечатать новую очередь.
2.4.5 Создать две очереди 10 случайных чисел в диапазоне от минус 40 до 40. Напечатать их. Поэлементно объединить их в одну очередь. Напечатать полученную очередь.
2.4.6 Список студентов группы (25 человек) организовать в виде стека. В записях элементов стека предусмотреть пол студента (М или Ж) – формировать случайным образом. Напечатать стек. Преобразовать этот стек в два по полу студентов. Напечатать новые стеки.
2.4.7 Создать стек 10 случайных чисел в диапазоне от минус 10 до 10. Напечатать его. Найти элемент стека с максимальным значением и поменять

его с первым элементом. Обмен выполнить с помощью указателей, а не «перезаписью» содержимого элементов. Напечатать новый стек.
2.4.8 Создать очередь 20 случайных чисел в диапазоне от 0 до 100. Напечатать ее. «Переписать» числа в новую очередь так, чтобы все четные числа разместились в начале очереди, а нечетные числа – в конце очереди. Напечатать новую очередь.
2.4.9 Алгебраическое выражение (без скобок) типа A*B+C*E+X, введенное в режиме диалога, посимвольно записывается в очередь. При очистке очереди определить число слагаемых в выражении.
2.4.10 Алгебраическое выражение (без скобок) типа A*B+C*E, введенное в режиме диалога, посимвольно записать в стек. При очистке стека определить число сомножителей в выражении.
2.4.11 Создать стек 20 случайных чисел в диапазоне от минус 20 до 20. Напечатать его. Найти элемент стека с минимальным значением и поменять его с последним элементом. Обмен выполнить с помощью указателей, а не «перезаписью» содержимого элементов. Напечатать новый стек.
2.4.12 Создать стек 20 случайных чисел в диапазоне от минус 50 до 50. Напечатать его. Поменять местами 1 элемент с 20 и 10 элемент с 11. Обмен выполнить с помощью указателей, а не «перезаписью» содержимого элементов. Напечатать новый стек.
2.4.13 Магазин за 1 день посещает 100 человек. Обслуживание покупателей предполагает продажу им : хлеба – число от 0 до 24; сыра – число от 25 до 49; печенья – число от 50 до 74; пиво – число от 75 до 99 из случайного числа в диапазоне от 0 до 100. Организовать очередь покупателей, а при ее обслуживании определить и напечатать какой продукт покупается чаще.
2.4.14 Произвольный текст, введенный в режиме диалога, посимвольно записать в стек. При очистке стека определить и напечатать сколько раз встречалась каждая буква.
2.4.15 Создать стек 20 случайных чисел в диапазоне от минус 100 до 100. Напечатать его. Найти элемент стека с максимальным значением и поменять его с элементом, имеющим минимальное значение. Обмен выполнить с помощью указателей, а не «перезаписью» содержимого элементов. Напечатать новый стек.
2.4.16 Создать стек 20 случайных чисел в диапазоне от 0 до 99. Напечатать его. «Переписать» числа в новый стек так, что все числа из одной цифры разместились в начале стека, а числа из двух цифр – в конце стека. Напечатать новый стек.
2.4.17 В автомобильной «пробке» застряло 200 авто – очередь. Марки авто: ВАЗ, Москвич и иномарка формируются случайным образом с равной вероятность. При обслуживании очереди (ликвидации «пробки») определить марка каких авто представлена в очереди максимально.
2.4.18 Создать стек 20 студентов, в каждый элемент которого включать номер студента и его баллы за экзамен. Баллы формировать случайным образом в диапазоне от 0 до 100. Напечатать стек. При очистке стека

определить и напечатать сколько студентов сдали экзамен на «отлично», «хорошо», «удовлетворительно» и «неудовлетворительно».
2.4.19 Создать стек 30 случайных чисел в диапазоне от 0 до 100. Напечатать его. Найти элемент стека с максимальным значением и поменять его с элементом, имеющим следующее максимальное значение. Обмен выполнить с помощью указателей, а не «перезаписью» содержимого элементов. Напечатать новый стек.
2.4.20 Создать очередь 10 случайных чисел в диапазоне от 0 до 100. Напечатать ее. Выполнить сортировку чисел в очереди в порядке убывания. Напечатать новую очередь.

2.5 Контрольные вопросы для защиты отчета на СРСП

2.5.1 Существующие формы представления информации? Примеры.
2.5.2 Какая форма представления информации используется в компьютерах?
2.5.3 Какая система счисления используется в компьютерах?
2.5.4 Какая форма представления чисел используется в памяти компьютера? Примеры.
2.5.5 Как записать шестнадцатеричное число BC в двоичной и десятичной формах?
2.5.6 Понятие поразрядных логических операций? Примеры.
2.5.7 Операции побитового сдвига? Примеры.
2.5.8 Понятие указателя? Примеры.
2.5.9 Работа с небезопасным кодом. Примеры.
2.5.10 Операции с указателями. Примеры.

ТЕМА 3 РЕГУЛЯРНЫЕ ВЫРАЖЕНИЯ В ЯЗЫКЕ C#

3.1 Цель лабораторной работы

Получить практические навыки по программированию регулярных выражений в языке C#.

3.2 Методические указания к лабораторной работе

Изучить материал 5 и 6 лекций дисциплины.

Изучить материал, рекомендуемый в силлабусе: стр. 355-365 [2]. – сокращенный вариант представлен:

Стандартный класс String позволяет выполнять над строками различные операции, в том числе поиск, замену, вставку и удаление подстрок. Тем не менее, есть классы задач по обработке символьной информации, где стандартных возможностей явно не хватает. Чтобы облегчить решение подобных задач, в Net Framework встроен более мощный аппарат работы со строками, основанный на регулярных выражениях. Специальное пространство имен RegularExpression, содержит набор классов,

обеспечивающих работу с регулярными выражениями. Полное описание, как теоретических основ, так и практических особенностей применения этого аппарата в C#, требует отдельной книги. Придется ограничиться конспектом лекций введением в эту интересную область работы со строками, не рассматривая подробно все классы, входящие в пространство имен RegularExpression.

3.3 Домашнее задание на лабораторную работу

Считать текст из файла и вывести его на экран монитора только предложения, состоящие из заданного количества слов (любых наборов символов между пробелами).

3.4 Индивидуальные задания для СРС

Дополнение 1. В некоторых индивидуальных заданиях необходимо выполнить анализ информации файла телефонных переговоров клиента, который среди прочей информации содержит записи об абонентах, включающих следующие сведения:

– номер абонента, с которым выполнялось соединение;
– дату и время соединения;
– продолжительность соединения;
– стоимость переговоров.

Дополнение 2. В некоторых индивидуальных заданиях необходимо выполнить анализ кода программы на наличие в нем различных элементов языка C#. Код программы должен быть представлен обычным файлом с расширением «.cs», в который Вы записываете необходимую информацию в соответствии с Вашим индивидуальным заданием.

Дополнение 3. В некоторых индивидуальных заданиях необходимо выполнить анализ некоторого текста. Текст, соответствующий Вашему заданию, должен быть помещен в текстовый файл с расширением «.txt» или «.doc».

3.4.1 В файле с телефонными переговорами клиента определить по какому номеру выполнено максимальное количество звонков.
3.4.2 Выполнить анализ кода программы на наличие в нем всех циклов for языка C#. Напечатать результаты анализа на экране монитора.
3.4.3 Считать текст из файла и вывести на экран монитора строку, содержащую максимальное количество знаков пунктуации.
3.4.4 В файле с телефонными переговорами клиента выполнить сортировку переговоров по их стоимости.
3.4.5 Выполнить анализ кода программы на наличие в нем комментариев языка C#. Напечатать результаты анализа на экране монитора.

3.4.6 Считать текст из файла и вывести его на экран монитора, заменив цифры словами, например, «0» на слово «ноль»; «1» на слово «один» и т.д.
3.4.7 В файле с телефонными переговорами клиента определить по какому номеру выполнено максимальное количество звонков.
3.4.8 Выполнить анализ кода программы на наличие в нем всех операторов присваивания языка C# (не учитывать записи типа for (i=0; i<=10; i++) …). Напечатать результаты анализа на экране монитора.
3.4.9 Считать текст из файла и вывести на экран монитора строку, содержащую максимальное количество чисел (не цифр).
3.4.10 В файле с телефонными переговорами клиента определить по какому номеру был самый продолжительный по времени разговор.
3.4.11 Выполнить анализ кода программы на наличие в нем всех операторов вывода информации на экран (консоль) языка C#. Напечатать результаты анализа на экране монитора.
3.4.12 Считать текст из файла и вывести его на экран монитора только цитаты текста, то есть предложения, заключенные в кавычки.
3.4.13 В файле с телефонными переговорами клиента определить по какому номеру выполнены подряд (2 и более соединений) звонки.
3.4.14 Выполнить анализ кода программы на наличие в нем всех операторов ввода информации с клавиатуры (консоли) языка C#. Напечатать результаты анализа на экране монитора.
3.4.15 Считать текст из файла и вывести на экран монитора строку, содержащую максимальное количество повторяющихся слов.
3.4.16 В файле с телефонными переговорами клиента определить по какому номеру выполнена максимальная непрерывная последовательность звонков.
3.4.17 Выполнить анализ кода программы на наличие в нем всех операторов условного перехода языка C#. Напечатать результаты анализа на экране монитора.
3.4.18 Считать текст из файла и вывести его на экран монитора только предложения, содержащие информацию о 1000 и более «у.е.».
3.4.19 В файле с телефонными переговорами клиента определить все номера, время связи с которыми было менее 5 секунд.
3.4.20 Выполнить анализ кода программы на наличие в нем всех циклов с условием, но не операторов if языка C#. Напечатать результаты анализа на экране монитора.

3.5 Контрольные вопросы для защиты отчета на СРСП

3.5.1 Понятие регулярного выражения. Примеры.
3.5.2 Основные задачи, решаемые с помощью регулярных выражений.
3.5.3 Символика языка регулярных выражений. Пример.
3.5.4 Понятие повторителей. Примеры.
3.5.5 Понятие метасимволов. Примеры.
3.5.6 Понятие заменителей или «Классов символов», Примеры.

3.5.7 Специальные (управляющие) символы. Примеры.
3.5.8 Методы регулярных выражений. Примеры.

ТЕМА 4 ПЕРЕДАЧА ДАННЫХ МЕЖДУ НИТЯМИ В ПРОЦЕССЕ НА ЯЗЫКЕ C#

4.1 Цель лабораторной работы

Получить практические навыки по созданию и использованию нитей для передачи данных в языке C#

4.2 Методические указания к лабораторной работе

Изучить материал 7, 8 и 9 лекций дисциплины.

Изучить материал, рекомендуемый в силлабусе: стр. 237-246 [2] – сокращенный вариант представлен:

« Многопоточные приложения

Приложение .NET состоит из одного или нескольких процессов. Процессу принадлежат выделенная для него область оперативной памяти и ресурсы. Каждый процесс может состоять из нескольких доменов (частей) приложения, ресурсы которых изолированы друг от друга. В рамках домена может быть запущено несколько потоков выполнения. Поток (thread) представляет собой часть исполняемого кода программы. В каждом процессе есть первичный поток, исполняющий роль точки входа в приложение. Для консольных приложений это метод Main.

Многопоточные приложения создают как для многопроцессорных, так и для однопроцессорных систем. Основной целью при этом являются повышение общей производительности и сокращение времени реакции приложения. Управление потоками осуществляет операционная система. Каждый поток получает некоторое количество квантов времени, по истечении которого управление передается другому потоку. Это создает у пользователя однопроцессорной машины впечатление одновременной работы нескольких потоков и позволяет, к примеру, выполнять ввод текста одновременно с длительной операцией по передаче данных.

Недостатки многопоточности:

- большое количество потоков ведет к увеличению накладных расходов, связанных с их переключением, что снижает общую производительность системы;
- в многопоточных приложениях возникают проблемы синхронизации данных, связанные с потенциальной возможностью доступа к одним и тем же данным со стороны нескольких потоков (например, если один поток начинает изменение общих данных, а отведенное ему время истекает, доступ к этим же данным может получить другой поток, который, изменяя данные, необратимо их повреждает)».

4.3 Домашнее задание на лабораторную работу

Написать программу, запускающую три различных дополнительных нити. Работу нитей контролировать в консольном окне монитора.

4.4 Индивидуальные задания для СРС

4.4.1 Разработать программу, в которой использовать две дополнительные нити – Н1 и Н2. Нить Н1 должна формировать и печатать 10 случайных чисел в диапазоне от 0 до 100, поочередно передавать их в нить Н2 и накапливать сумму этих чисел. Нить Н2 должна получать числа из Н1, печатать их, вычислять сумму полученных чисел и итоговую сумму передавать нити Н1. Нить Н1 должна сравнить сумму полученную из Н2 и с ее накопленной суммой. По результатам сравнения напечатать соответствующие комментарии.

4.4.2 Разработать программу, в которой использовать три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в цикле формирует и печатает массив 10 случайных чисел в диапазоне от 0 до 100 и передает его нити Н2. Нить Н2 сдвигает полученный массив на два разряда влево, печатает новый массив и передает его нити Н3. Нить Н3 сдвигает полученный массив на два разряда вправо, печатает его и передает его нити Н1. Нить Н1 сравнивает исходный массив с массивом полученным от нити Н3. По результатам сравнения печатаются соответствующие комментарии.

4.4.3 Разработать программу, в которой для решения квадратного уравнения используются три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в режиме диалога получает значения коэффициентов квадратного уравнения a, b, c и передает их в нить Н2. Нить Н2 определяет, существуют ли действительные корни уравнения и значения дискриминанта передает нити Н3 с одновременной печатью необходимых комментарий. Нить Н3 вычисляет и печатает действительные корни квадратного уравнения.

4.4.4 Разработать программу, в которой использовать три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в цикле формирует и печатает матрицу 5х5 случайных чисел в диапазоне от 0 до 70 и передает его нити Н2. Нить Н2 трансформирует полученную матрицу, печатает ее и передает нити Н3. Нить Н3 повторно трансформирует полученную матрицу, печатает ее и передает нити Н1. Нить Н1 сравнивает исходную матрицу с матрицей полученной от нити Н3. По результатам сравнения печатаются соответствующие комментарии.

4.4.5 Разработать программу, в которой использовать две дополнительные нити – Н1 и Н2. Нити Н1 и Н2 содержат циклы for от 1 до 10, печатают значения управляющих переменных и поочередно передают их в нити Н2 и Н1 соответственно. Каждая нить, получившая значение управляющей переменной увеличивает его на 1 и сравнивает с текущим значением. По результатам сравнения напечатать соответствующие комментарии.

4.4.6 Разработать программу, в которой использовать три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в цикле формирует и печатает массив 10 случайных чисел в диапазоне от 0 до 100 и передает его нитям Н2 и Н3. Нить Н2 находит и печатает максимальный элемент полученного массива. Нить Н3 сортирует и печатает новый массив.
4.4.7 Разработать программу, в которой использовать три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в цикле формирует и печатает массив 10 случайных чисел в диапазоне от 0 до 10 и передает его нитям Н2 и Н3. Нить Н2 вычисляет значение синуса и косинуса для каждого полученного значения, печатает их и нити Н3. Нить Н3 для вычисляет значение тангенса для каждого полученного значения из нити Н1 и сравнивает его с результатом деления синуса на косинус, полученных из нити Н2. По результатам сравнения печатаются соответствующие комментарии.
4.4.8 Разработать программу, в которой использовать три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в цикле формирует и печатает матрицу 6х6 случайных чисел в диапазоне от минус 40 до 40 и передает его нити Н2. Нить Н2 сортирует по строкам в порядке убывания полученную матрицу, печатает ее и передает нити Н3. Нить Н3 сортирует по строкам в порядке возрастания полученную матрицу, печатает ее и передает нити Н1. Нить Н1 вычисляет суммы каждой строки исходной матрицы и полученной от нити 3 и сравнивает их. По результатам сравнения печатаются соответствующие комментарии.
4.4.9 Разработать программу, в которой использовать две дополнительные нити – Н1 и Н2. Нити Н1 и Н2 содержат циклы for от 1 до 10. На каждом шаге цикла Н1 передает нити основной программе символ 1 и печатает его на экране, а Н2 передает нити основной программе символ 0 и также печатает его на экране. Нить основной программы получает комбинации символов «101010 . . .» и по окончании работы нитей Н1 и Н2 сравнивает полученную комбинацию символов с напечатанной. По результатам сравнения напечатать соответствующие комментарии.
4.4.10 Разработать программу, в которой использовать две дополнительные нити – Н1 и Н2. Нить Н1 в цикле формирует и печатает массив 10 случайных чисел в диапазоне от минус 50 до 50 и передает его нити Н2. Нить Н2 находит и печатает максимальный и минимальный элементы полученного массива и индексы найденных элементов передает нити Н1. Нить Н1 выполняет обмен элементов по полученным индексам и печатает новый массив.
4.4.11 Разработать программу, в которой использовать три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в цикле должна формировать и печатать 10 случайных чисел в диапазоне от минус 50 до 50, поочередно передавать их в нити Н2 и Н3. Нить Н2 должна получить числа из Н1 вычислить, напечатать и вернуть сумму только положительных чисел. Нить Н3 должна получить числа из Н1 вычислить, напечатать и вернуть сумму только отрицательных чисел. Нить Н1 должна сравнить сумму всех сформированных чисел с

суммой, полученной от нитей Н2 и Н3. По результатам сравнения на каждом шаге циклов печатаются соответствующие комментарии.

4.4.12 Разработать программу, в которой использовать две дополнительные нити – Н1 и Н2. Нить Н1 в цикле формирует и печатает матрицу 6х6 случайных чисел в диапазоне от 0 до 100 и передает его нити Н2. Нить Н2 формирует и печатает массив максимальных значение каждой строки матрицы.

4.4.13 Разработать программу, в которой использовать две дополнительные нити – Н1 и Н2. Нить Н1 содержит цикл for от 0 до 10, а нить Н2 цикл от 10 до 0. На каждом шаге цикла Н1 и Н2 передают нити основной программы свои значения управляющих переменных циклов. Нить основной программы суммирует полученные значения и сравнивает с числом 10. По результатам сравнения на каждом шаге циклов печатаются соответствующие комментарии.

4.4.14 Разработать программу, в которой использовать три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в цикле формирует и печатает массив 10 случайных чисел в диапазоне от минус 30 до 30 и передает его нити Н2. Нить Н2 «переворачивает» полученный массив, печатает новый массив и передает его нити Н3. Нить Н3 «переворачивает» полученный массив, печатает его и передает его нити Н1. Нить Н1 сравнивает исходный массив с массивом полученным от нити Н3. По результатам сравнения печатаются соответствующие комментарии.

4.4.15 Разработать программу, в которой для вычисления площади прямоугольника используются три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в режиме диалога получает значения двух сторон прямоугольника a и b и передает их в нить Н2. Нить Н2 определяет, является ли заданный прямоугольник квадратом и если да, то вычисляет и печатает площадь квадрата. Если заданный прямоугольник не является квадратом, то нить Н2 передает значение сторон нити Н3, которая вычисляет и печатает площадь прямоугольника. Запрещается одновременная печать и квадрата и квадратного прямоугольника.

4.4.16 Разработать программу, в которой использовать три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в цикле формирует и печатает матрицу 5х5 случайных чисел в диапазоне от минус 50 до 50 и передает его нити Н2. Нить Н2 сортирует по столбцам в порядке возрастания полученную матрицу, печатает ее и передает нити Н3. Нить Н3 сортирует по столбцам в порядке убывания полученную матрицу, печатает ее и передает нити Н1. Нить Н1 вычисляет суммы каждого столбца исходной матрицы и полученной от нити 3 и сравнивает их. По результатам сравнения печатаются соответствующие комментарии.

4.4.17 Разработать программу, в которой использовать три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в цикле должна формировать и печатать 10 случайных чисел в диапазоне от 0 до 100, поочередно передавать их в нить Н2. Нить Н2 должна получать числа из Н1, печатать их и передавать нити Н3. Нить Н3 должна получать числа из Н2, печатать их и передавать нити

Н1. Нить Н1 должна сравнить число, полученное от нити Н3 с числом, отправленным нити Н2. По результатам сравнения на каждом шаге циклов печатаются соответствующие комментарии.

4.4.18 Разработать программу, в которой использовать три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в цикле формирует и печатает массив 10 случайных чисел в диапазоне от минус 40 до 40 и передает его нитям Н2 и Н3. Нить Н2 сортирует только положительные элементы массива и печатает новый массив. Нить Н3 сортирует только отрицательные элементы массива и печатает новый массив.

4.4.19 Разработать программу, в которой использовать три дополнительные нити – Н1, Н2 и Н3. Нить Н1 в цикле формирует и печатает массив 10 случайных чисел в диапазоне от 0 до 100 (радиусов кругов) и передает его нитям Н2 и Н3. Нить Н2 вычисляет и печатает площадь круга с максимальным радиусом, которая возвращается в нить Н1. Нить Н3 вычисляет и печатает площадь круга с минимальным радиусом, которая возвращается в нить Н1. Нить Н1 вычисляет и печатает сколько кругов (по площади), полученных из нити Н3, может разместиться в круге, полученном из нити Н2.

4.4.20 Разработать программу, в которой использовать две дополнительные нити – Н1 и Н2. Нить Н1 в цикле формирует и печатает матрицу 6х6 случайных чисел в диапазоне от 0 до 100 и передает его нити Н2. Нить Н2 формирует и печатает массив минимальных значений каждого столбца матрицы.

4.5 Контрольные вопросы для защиты отчета на СРСП

4.5.1 Понятие нити? Примеры.

4.5.2 Основные состояния компьютера и нити? Примеры.

4.5.3 Как называется программа, распределяющая время работы процессора между нитями?

4.5.4 Основные операции работы с нитью? Примеры.

4.5.5 В каком случае метод называется безопасным с точки зрения выполнения нити?

4.5.6 Основные свойства нити? Примеры.

4.5.7 Что передается конструктору класса Thread при создании объекта нити?

4.5.8 Имеется один объект некоторого класса, содержащего метод вывода чисел в консольное окно и локальную переменную, управляющую выводом чисел (разрешить – не разрешить). Как будут работать с этим методом главная и одна дополнительная нити, если каждая нить меняет значение локальной переменной на противоположное значение?

4.5.9 Понятие фоновой нити приложения?

4.5.10 Понятие приоритета нити? Примеры.

ТЕМА 5 СИНХРОНИЗАЦИЯ РАБОТЫ НИТЕЙ В ЯЗЫКЕ C#

5.1 Цель лабораторной работы

Получить практические навыки по синхронизации нитей процессов компьютера.

5.2 Методические указания к лабораторной работе

Изучить материал 10, 11 и 12 лекций дисциплины.

Изучить материал, рекомендуемый в силлабусе: стр. 739-793 [3] – сокращенный вариант представлен:

« Блокировка сама по себе очень быстра: она требует десятков наносекунд, если собственно блокирования не происходит. Если требуется блокирование, то последующее переключение задач занимает уже микросекунды или даже миллисекунды на перепланировку потоков. Однако сравните это с часами, которые вы должны будете потратить, не поставив lock там, где надо.

При неправильном использовании у блокировки могут быть и негативные последствия – уменьшение возможности параллельного исполнения потоков, взаимоблокировки, гонки блокировок. Возможности для параллельного исполнения уменьшаются, когда слишком много кода помещено в конструкцию lock, заставляя другие потоки простаивать все время, пока этот код исполняется. Взаимоблокировка наступает, когда каждый из двух потоков ожидает на блокировке другого потока и, таким образом, ни тот, ни другой не могут двинуться дальше. Гонкой блокировок называется ситуация, когда любой из двух потоков может первым получить блокировку, однако программа ломается, если первым это сделает “неправильный” поток.

Взаимоблокировка – общий синдром многих объектов синхронизации. Хорошее правило, помогающее избегать взаимоблокировок, состоит в том, чтобы начинать с блокировки минимального количества объектов, и увеличивать степень детализации блокировок, когда размер кода в блокировке чрезмерно увеличивается.»

5.3 Домашнее задание на лабораторную работу

Синхронизировать три нити одного процесса с помощью оператор lock. Каждая нить поочередно формирует и печатает три числа a, b и c. Числа первой нити формируются в диапазоне от 10 до 19, второй нити – от 20 до 29 и третьей нити – от 30 до 39. Печать чисел каждой нитью производится по диагонали сверху вниз и слева направо (печать каждой нити занимает три строки) в продолжение печати предыдущей нити.

5.4 Индивидуальные задания для СРС

5.4.1 Синхронизировать три нити одного процесса с помощью глобальных переменных. Каждая нить формирует 3 числа для одноименной строки

матрицы 3x3. Числа первой нити формируются в диапазоне от 10 до 19, второй нити – от 20 до 29 и третьей нити – от 30 до 39. Процесс управляет всеми нитями и выводит на экран результат их работы.

5.4.2 Синхронизировать три нити одного процесса с помощью метода Sleep. Первая нить выводит на экран монитора имя студента, вторая нить – его отчество и третья нить – его фамилию. Вывод должен выполняться в указанном порядке. Процесс синхронизирует работу всех нитей.

5.4.3 Синхронизировать три нити одного процесса с помощью метода Join. Первая нить выводит на экран монитора текущее время в минутах, вторая нить – текущее время в секундах, а третья нить – текущее время в миллисекундах. Вывод должен выполняться в указанном порядке. Процесс синхронизирует работу всех нитей.

5.4.4 Синхронизировать три нити одного процесса с помощью метода Join. Каждая нить формирует 3 числа для одноименной строки матрицы 3x3. Числа первой нити формируются в диапазоне от 10 до 19, второй нити – от 20 до 29 и третьей нити – от 30 до 39. Процесс управляет всеми нитями и выводит на экран результат их работы.

5.4.5 Синхронизировать три нити одного процесса с помощью оператора lock. Каждая нить выводит на экран монитора в заданном порядке три слова некоторого известного предложения. Процесс синхронизирует работу всех нитей.

5.4.6 Синхронизировать три нити одного процесса с помощью глобальных переменных. Первая нить выводит на экран монитора текущий месяц, вторая нить – день, а третья нить – час работы программы. Вывод должен выполняться в указанном порядке. Процесс синхронизирует работу всех нитей.

5.4.7 Синхронизировать три нити одного процесса с помощью метода Sleep. Каждая нить формирует 4 числа для одноименной строки матрицы 3x4. Числа первой нити формируются в диапазоне от 10 до 19, второй нити – от 20 до 29 и третьей нити – от 30 до 39. Процесс управляет всеми нитями и выводит на экран результат их работы.

5.4.8 Синхронизировать три нити одного процесса с помощью метода Join. Каждая нить выводит на экран монитора одно слово некоторого известного предложения. Процесс синхронизирует работу всех нитей.

5.4.9 Синхронизировать три нити одного процесса с помощью метода Sleep. Первая нить выводит на экран монитора текущее время в часах, вторая нить – текущее время в минутах, а третья нить – текущее время в секундах. Вывод должен выполняться в указанном порядке. Процесс синхронизирует работу всех нитей.

5.4.10 Синхронизировать три нити одного процесса с помощью оператора lock. Каждая нить формирует 5 чисел для одноименной строки матрицы 3x5. Числа первой нити формируются в диапазоне от 10 до 19, второй нити – от 20 до 29 и третьей нити – от 30 до 39. Процесс управляет всеми нитями и выводит на экран результат их работы.

5.4.11 Синхронизировать три нити трех процессов с помощью мьютекса. Первая нить первого процесса выводит на экран монитора фамилию студента, первая нить второго процесса – его имя и первая нить третьего процесса – его отчество.
5.4.12 Синхронизировать три нити одного процесса с помощью оператора lock. Первая нить выводит на экран монитора текущий месяц, вторая нить – день, а третья нить – час работы программы. Вывод должен выполняться в указанном порядке. Процесс синхронизирует работу всех нитей.
5.4.13 Синхронизировать три нити одного процесса с помощью автоматической синхронизации. Каждая нить формирует 3 числа для одноименной строки матрицы 3x3. Числа первой нити формируются в диапазоне от 10 до 19, второй нити – от 20 до 29 и третьей нити – от 30 до 39. Процесс управляет всеми нитями и выводит на экран результат их работы.
5.4.14 Синхронизировать три нити трех процессов с помощью семафора. Первая нить первого процесса выводит на экран монитора фамилию студента, первая нить второго процесса его оценки по математике и физике и первая нить третьего процесса его оценку по системному программированию.
5.4.15 Синхронизировать три нити трех процессов с помощью мьютекса. Первая нить первого процесса выводит на экран монитора текущее время в часах, первая нить второго процесса – текущее время в минутах, а первая нить третьего процесса – текущее время в секундах.
5.4.16 Синхронизировать три нити трех процессов с помощью мьютекса. Каждый процесс формирует 3 числа для одноименной строки матрицы 3x3. Первая нить первого процесса формирует и выводит на экран монитора числа от 10 до 19, первая нить второго процесса – от 20 до 29 и первая нить третьего процесса – от 30 до 39.
5.4.17 Синхронизировать три нити одного процесса с помощью глобальных переменных. Каждая нить выводит на экран монитора два слова некоторого известного предложения. Процесс синхронизирует работу всех нитей.
5.4.18 Синхронизировать три нити трех процессов с помощью семафора. Первая нить первого процесса выводит на экран монитора сторону треугольника A, первая нить второго процесса – сторону треугольника B, а первая нить третьего процесса – сторону треугольника C. Первый процесс определяет можно ли из полученных значений сторон построить треугольник и печатает в консольном окне соответствующие комментарии.
5.4.19 Синхронизировать три нити трех процессов с помощью семафора. Каждый процесс формирует 4 числа для одноименной строки матрицы 3x4. Первая нить первого процесса формирует и выводит на экран монитора числа от 10 до 19, первая нить второго процесса – от 20 до 29 и первая нить третьего процесса – от 30 до 39.
5.4.20 Синхронизировать три нити одного процесса с помощью автоматической синхронизации. Каждая нить выводит на экран монитора два слова некоторого известного предложения. Процесс синхронизирует работу всех нитей.

5.5 Контрольные вопросы для защиты отчета на СРСП

5.5.1 Понятие действия?
5.5.2 Понятие синхронизации нитей?
5.5.3 Классификация средств синхронизации нитей?
5.5.4 Как с помощью глобальной переменно можно синхронизировать работу нитей одного процесса?
5.5.5 Как с помощью метода Sleep можно синхронизировать работу нитей одного процесса?
5.5.6 Как с помощью метода Join можно синхронизировать работу нитей одного процесса?
5.5.7 Как с помощью оператора lock можно синхронизировать работу нитей одного процесса?
5.5.8 Как с помощью объекта синхронизации Mutex можно синхронизировать работу нитей?
5.5.9 Как с помощью объекта синхронизации Semaphore можно синхронизировать работу нитей?
5.5.10 Понятие автоматической синхронизации нитей.

ТЕМА 6 ИСПОЛЬЗОВАНИЕ КАНАЛОВ ПЕРЕДАЧИ ДАННЫХ В ЯЗЫКЕ C#

6.1 Цель лабораторной работы

Получить практические навыки по организации обмена данными помощью каналов между различными процессами как на одном компьютере, так и в локальной сети.

6.2 Методические указания к лабораторной работе

Изучить материал 13, 14 и 15 лекций дисциплины

Изучить материал, рекомендуемый в силлабусе: стр. 495-535 [3] – сокращенный вариант представлен:

«Именованные каналы – это механизм межпроцессорной коммуникации, который позволяет процессам обмениваться данными. Каждый канал создаваемый сервером может принимать множество клиентских соединений. После того как соединения установлено, и сервер и клиент могут взаимодействовать, используя обычные механизмы потоков .NET Framework. Вы должны использовать одинаковое имя для каналов сервера и клиента».

6.3 Домашнее задание на лабораторную работу

В качестве домашнего задания разработать программы для двух процессов с одним текстовым файлом, через который эти программы обмениваются текстовыми сообщениями.

6.4 Индивидуальные задания для СРС

6.4.1 Сервер создает и печатает массива 5 целых случайных чисел в диапазоне от 0 до 100. С помощью файла сервер передает созданный массив клиенту. Клиент принимает массив, выполняет сортировку в порядке убывания чисел, печатает его и возвращает серверу. Сервер принимает отсортированный массив и печатает его.
6.4.2 Сервер создает и печатает массива 15 вещественных случайных чисел в диапазоне от 0 до 40. С помощью именованного канала передачи сервер передает созданный массив клиенту. Клиент принимает массив, выполняет поиск двух максимальных чисел, печатает их и возвращает серверу. Сервер принимает найденные максимальные числа и печатает их.
6.4.3 Сервер создает и печатает массива 20 байтов, сформированных случайным образом в диапазоне от 0 до 255. С помощью анонимного канала передачи сервер передает созданный массив клиенту. Клиент принимает массив, выполняет поиск байтов с наибольшим и наименьшим количеством двоичных единиц, печатает их и возвращает серверу. Сервер принимает найденные байты и печатает их.
6.4.4 Сервер создает и печатает массива 10 символов, сформированных случайным образом в диапазоне латинских букв. С помощью файла сервер передает созданный массив клиенту. Клиент принимает массив, выполняет поиск гласных букв, печатает их и возвращает серверу. Сервер принимает найденные буквы и печатает их.
6.4.5 Сервер создает и печатает матрицу 8x3 логических переменных, сформированных случайным образом. С помощью именованного канала передачи сервер передает созданный матрицу клиенту. Клиент принимает матрицу, выполняет поиск всех строк матрицы, в которых все переменные true, печатает номера этих строк и возвращает их серверу. Сервер принимает найденные номера строк матрицы и печатает их.
6.4.6 Сервер вводит в режиме диалога некоторое сообщение строкового типа и печатает его. С помощью анонимного канала передачи сервер передает полученное сообщение клиенту. Клиент принимает сообщение, выполняет поиск максимального и минимального слов сообщения, печатает их и возвращает серверу. Сервер принимает найденные слова сообщения и печатает их.
6.4.7 Сервер считывает с диска документ в формате XML и печатает его. С помощью именованного канала передачи сервер передает полученный документ клиенту. Клиент принимает документ, печатает его и возвращает серверу. Сервер принимает документ от клиента и повторно печатает его.
6.4.8 Сервер создает и печатает массива 8 целых случайных чисел в диапазоне от минус 40 до 40. С помощью анонимного канала передачи сервер передает созданный массив клиенту. Клиент принимает массив,

выполняет сортировку в порядке убывания чисел, печатает его и возвращает серверу. Сервер принимает отсортированный массив и печатает его.
6.4.9 Сервер создает и печатает массива 10 вещественных случайных чисел в диапазоне от 0 до 10. С помощью файла сервер передает созданный массив клиенту. Клиент принимает массив, выполняет поиск двух максимальных чисел, печатает их и возвращает серверу. Сервер принимает найденные максимальные числа и печатает их.
6.4.10 Сервер создает и печатает массива 15 байтов, сформированных случайным образом в диапазоне от 0 до 255. С помощью именованного канала передачи сервер передает созданный массив клиенту. Клиент принимает массив, выполняет поиск байтов с наибольшим и наименьшим количеством двоичных единиц, печатает их и возвращает серверу. Сервер принимает найденные байты и печатает их.
6.4.11 Сервер создает и печатает массива 20 символов, сформированных случайным образом в диапазоне латинских букв. С помощью анонимного канала передачи сервер передает созданный массив клиенту. Клиент принимает массив, выполняет поиск гласных букв, печатает их и возвращает серверу. Сервер принимает найденные буквы и печатает их.
6.4.12 Сервер создает и печатает матрицу 6x3 логических переменных, сформированных случайным образом. С помощью файла сервер передает созданный матрицу клиенту. Клиент принимает матрицу, выполняет поиск всех строк матрицы, в которых все переменные true, печатает номера этих строк и возвращает их серверу. Сервер принимает найденные номера строк матрицы и печатает их.
6.4.13 Сервер вводит в режиме диалога некоторое сообщение строкового типа и печатает его. С помощью именованного канала передачи сервер передает полученное сообщение клиенту. Клиент принимает сообщение, выполняет поиск максимального и минимального слов сообщения, печатает их и возвращает серверу. Сервер принимает найденные слова сообщения и печатает их.
6.4.14 Сервер считывает с диска документ в формате XML и печатает его. С помощью анонимного канала передачи сервер передает полученный документ клиенту. Клиент принимает документ, печатает его и возвращает серверу. Сервер принимает документ от клиента и повторно печатает его.
6.4.15 Сервер создает и печатает массива 10 целых случайных чисел в диапазоне от минус 50 до 50. С помощью именованного канала передачи сервер передает созданный массив клиенту. Клиент принимает массив, выполняет сортировку в порядке убывания чисел, печатает его и возвращает серверу. Сервер принимает отсортированный массив и печатает его.
6.4.16 Сервер создает и печатает массива 20 вещественных случайных чисел в диапазоне от 0 до 80. С помощью анонимного канала передачи сервер передает созданный массив клиенту. Клиент принимает массив, выполняет поиск двух максимальных чисел, печатает их и возвращает серверу. Сервер принимает найденные максимальные числа и печатает их.

6.4.17 Сервер создает и печатает массива 10 байтов, сформированных случайным образом в диапазоне от 0 до 255. С помощью файла сервер передает созданный массив клиенту. Клиент принимает массив, выполняет поиск байтов с наибольшим и наименьшим количеством двоичных единиц, печатает их и возвращает серверу. Сервер принимает найденные байты и печатает их.
6.4.18 Сервер создает и печатает массива 15 символов, сформированных случайным образом в диапазоне латинских букв. С помощью именованного канала передачи сервер передает созданный массив клиенту. Клиент принимает массив, выполняет поиск гласных букв, печатает их и возвращает серверу. Сервер принимает найденные буквы и печатает их.
6.4.19 Сервер создает и печатает матрицу 10x3 логических переменных, сформированных случайным образом. С помощью анонимного канала передачи сервер передает созданный матрицу клиенту. Клиент принимает матрицу, выполняет поиск всех строк матрицы, в которых все переменные true, печатает номера этих строк и возвращает их серверу. Сервер принимает найденные номера строк матрицы и печатает их.
6.4.20 Сервер вводит в режиме диалога некоторое сообщение строкового типа и печатает его. С помощью файла сервер передает полученное сообщение клиенту. Клиент принимает сообщение, выполняет поиск максимального и минимального слов сообщения, печатает их и возвращает серверу. Сервер принимает найденные слова сообщения и печатает их.

6.5 Контрольные вопросы для защиты отчета на СРСП

6.5.1 Способы передачи данных между процессами.
6.5.2 Связи между процессами.
6.5.3 Передача сообщений.
6.5.4 Синхронный и асинхронный обмен данными.
6.5.5 Обмен данными между процессами с помощью файла.
6.5.6 Понятие именованного канала связи.
6.5.7 Порядок работы с именованными каналами связи.
6.5.8 Понятие потокового адаптера.
6.5.9 Понятие анонимного канала передачи данных.
6.5.10 Использование анонимного канала для передачи текстовой информации.

7 СПИСОК ЛИТЕРАТУРЫ

7.1. Основная литература

7.1.1 Презентации лекций по дисциплине «Системное программирование» для студентов специальности 5В0704 – смотри портал кафедры ИС http : \\ www.do.ektu.kz
7.1.2 В.В. Фаронов «Создание приложений с помощью C#» Руководство программиста. - М.: “Эксмо”, 2008г.
7.1.3 Т.А. Павловская C#, Программирование на языке высокого уровня. Учебник для вузов, СПб,: Питер, 2009г.
7.1.4 Д. Албахари, Б. Албахари «C# 3.0 справочник» СПб,:«БХВ - Петербург» 2009г.
7.1.5 В.М. Рябенький и др. Компьютерное управление внешними устройствами через стандартные интерфейсы, Учебное пособие, Олди-плюс, Херсон, 2008г.

7.2 Дополнительная литература

7.2.1 Э. Йодан Структурное программирование и конструирование программ. М.: ”Мир”, 1989г.
7.2.2 Н. Вирт Алгоритмы и структуры данных. М. Изд-во «МИР», 1989г.
7.2.3 Э.Троелсен C# и платформа .NET Библиотека программиста, СПб,:Питер, 2007г.

8 ПРИЛОЖЕНИЕ

При оформлении отчета по лабораторной работе рекомендуется следующая структура и последовательность элементов:
– титульный лист;
– название лабораторной работы;
– цель лабораторной работы;
– индивидуальное задание на лабораторную работу;
– краткие комментарии по выполнению индивидуального задания (при необходимости структурная схема алгоритма решения задачи);
– необходимый программный код индивидуального задания;
– результаты работы программы;
– выводы.

Титульный лист является первой страницей отчета и служит источником информации, необходимой для поиска документа. Поэтому он содержит название министерства, название университета, название кафедры, название дисциплины, название модуля дисциплины, ФИО студента, выполнившего лабораторную работу и ФИО преподавателя, принимающего отчет по лабораторной работе. Внизу титульного листа указывается место и год выполнения лабораторной работы, например, Усть–Каменогорск 2010г.

Титульный лист включают в общую нумерацию страниц отчета, но номер страницы на титульном листе не проставляется.

Название лабораторной работы должно соответствовать названию из методических указаний по выполнению лабораторных работ, часто оно соответствует названию модуля.

Цель лабораторной работы содержит краткое описание цели соответствующего модуля дисциплины.

Индивидуальное задание на лабораторную работу содержит полный текст индивидуального задания, полученного у преподавателя только после выполнения домашнего задания по лабораторной работе.

Краткие комментарии по выполнению индивидуального задания содержат описание алгоритма выполнения индивидуального задания. При необходимости приводится структурная схема алгоритма или его подробное словесное описание.

Необходимый программный код индивидуального задания содержит либо полный текст кода программы, либо фрагменты кода программы, разработанные студентом и коды добавляемые средой программирования, без которых объяснение выполненной работы невозможно.

Результаты работы программы обычно содержат копии окон работы программы во всех ее режимах.

В выводах обычно отмечается результат выполнения лабораторной работы.

Страницы текста отчета должны соответствовать формату А4.

Печатание отчета выполняется машинописным способом или с применением печатающих и графических устройств вывода ЭВМ на одной стороне листа белой бумаги. Тип шрифта - Times New Roman, основной размер шрифта - № 14, допускается № 12. Основной интервал -1, допускается -1,5.

Текст отчета следует печатать, соблюдая следующие разделы полей: правое, верхнее, нижнее и левое – 20 мм.

Абзацный отступ начинается не менее чем с четвертого знака и должен быть одинаков в пределах одного документа.

Вне зависимости от способа выполнения отчета, качество напечатанного текста, иллюстраций, таблиц и приложений должно удовлетворять требованию их чёткого воспроизведения.

Вписывать в отпечатанный текст отчета отдельные слова, формулы и знаки допускается только черными чернилами или черной тушью, при этом плотность вписанного текста должна быть максимально приближена к плотности основного текста.

Опечатки, описки и графические неточности допускается исправлять подчисткой или закрашиванием белой краской и нанесением на том же месте исправленного изображения машинописным способом или от руки черными чернилами лили чёрной тушью.

Повреждения листов отчета, помарки и следы не полностью удалённого текста не допускаются.

Разделы должны иметь порядковые номера в пределах всего отчета, обозначенные арабскими цифрами без точки и записанные с отступом.

Подразделы должны иметь нумерацию в пределах каждого раздела. Номер подраздела состоит из номеров раздела и подраздела, разделённых точкой. В конце номера подраздела точки не ставятся. Разделы, как и подразделы, могут состоять из одного или нескольких пунктов.

Разделы, подразделы должны иметь заголовки. Пункты заголовков не имеют.

Заголовки разделов документа следует располагать в середине строки без точки в конце и печатать прописными буквами, не подчёркивая и не выделяя.

Заголовки подразделов документа следует располагать в середине строки и печатать строчными буквами, начиная с первой прописной, выделяя жирным шрифтом.

Если раздел не имеет подразделов, то нумерация пунктов в нем должна быть в пределах этого раздела, и номер пункта должен состоять из номеров раздела и пункта, разделённых точкой. В конце номера пункта точка не ставится.

Если раздел имеет подразделы, то нумерация пунктов должна быть в пределах каждого подраздела и номер пункта должен состоять из номеров раздела, подраздела и пункта, разделённых точками.

Страницы отчета следует нумеровать арабскими цифрами, соблюдая сквозную нумерацию по всему тексту.

Номер страницы отчета проставляют по центру страницы вверху без точки в конце.

Страницы отчета скрепляются скрепкой или размещаются в файле (степлер не использовать).