План лабораторных работ по ОАиП на II семестр (Гуревич, 2019-20 учебный год)

1. Структуры
2. Файлы
3. Рекурсия
4. Сортировка (QuickSort)
5. Односвязные списки (стеки)
6. Двусвязные списки (очереди)
7. Нелинейные списки (деревья)
8. Графика
9. Решение нелинейных уравнений (поиск корней)
10. Побитовые операции (см. ниже подробности)

Примечания:

Работы можно выполнять как в консольном, так и в оконном режиме. Вариант работы – как в первом семестре (кроме л/р «Рекурсия», «Графика» и «Решение нелинейных уравнений», см. ниже).

Порядок сдачи лабораторных работ значения не имеет; приведенный порядок ориентировочно соответствует порядку лекций.

№1-2. Структуры и файлы:

Л/р №8 из методички (часть 1).

Большинству студентов рекомендуется сначала сделать ее без использования файлов, и это засчитывается как лабораторная работа «Структуры».

Затем они делают ее же (тот же вариант) с возможностью сохранения в файл и чтения из него (тип файла – любой, бинарный либо текстовый, по желанию студента). При этом нужно создать меню (в консоли), либо несколько кнопок или экранное меню (в оконном режиме): «Сохранить», «Загрузить» и т.д. Это уже засчитывается как лабораторная работа «Файлы».

Однако при желании студент может делать эту работу сразу с файлами, и она будет засчитана как обе лабораторные работы №1 и №2. Но в этом случае при защите он должен отвечать на вопросы как по структурам, так и по файлам.

Примечание 1: практика предыдущих лет показала, что защита лаб по теме «Файлы» для многих студентов оказывалась трудной. Это было связано с выполнением программ по образцам без детального понимания их работы. Перед защитой убедитесь, что понимаете смысл каждого из тех параметров функций работы с файлами, которые используются в Вашей программе.

Примечание 2: студентам, претендующим на отличную оценку, могут быть даны дополнительные лабораторные задания по теме «Файлы». Все эти дополнительные задания будут выполняться по тем же правилам, что и обычные лабы (т.е. их можно будет делать и дома). Условия этих заданий будут выданы после лекции по теме «Файлы»; поэтому тем студентам, которые сдадут лабораторную работу «Файлы» до этого времени (до конца февраля), такие дополнительные задания даваться не будут.

№3. Рекурсия:

Лабораторная работа №1 из методички (Часть 2).

Задача должна быть решена 2 способами (рекурсивным, и любым не рекурсивным). В варианте №5 допускается реализация только рекурсивного способа.

Студенты могут сами выбирать вариант задачи (т.к. в некоторых вариантах условие трудно для понимания).

№4. Сортировка:

Л/р №9 из методички (часть 2).

Цель работы – закрепить навыки работы со структурами, а также освоить алгоритмы QuickSort и поиска делением пополам.

Внимание! В методичке на сайте БГУИР (2009 г.) в примере функции Quick_Sort есть опечатка, из-за которой она не всегда сортирует верно! Она состоит в том, что переменная x используется для двух разных целей: хранения ключа и обмена. Для исправления заведите 2 разные переменные (например, x для ключа и X для обмена). В лабораторной работе у этих переменных могут оказаться даже разные типы.

№5, №6. Связанные списки:

Л/р №2 и №3 из методички (часть 2).

Рисовать блок-схемы не требуется.

Студенты, не претендующие на отличную оценку, могут не делать сортировку односвязного списка.

№7. Деревья:

Л/р №5 из методички (часть 2).

Рисовать блок-схемы не требуется.

Студенты, не претендующие на отличную оценку, могут опустить часть «общих» (не зависящих от варианта) пунктов.

Но в любом случае должны присутствовать построение дерева, вывод его на экран, добавление в него элементов и выполнение индивидуального задания.

Внимание! В методичке на сайте БГУИР (2009 г.) в примере функции Del_Info содержится ошибка, из-за которой она не всегда правильно удаляет узел! Для исправления надо оператор R->left = Prev_R; *заменить на* R->left = Del->left;

№8. Графика:

Теория, которая будет дана на лекциях:

- загрузка картинок из файлов (например, через компонент Image, свойство Picture);
- рисование фигур – например:

```
Image1->Canvas->MoveTo(20,20);
Image1->Canvas->LineTo(20,100);
```

- а также рисование графиков в компоненте Chart.

Компоненты Image и Chart в C++ Builder находятся на вкладке Additional.

Если в цикле поочередно выводятся картинки (для создания анимации), они могут не отображаться на экране сразу же. Для устранения этого служит, например, функция

```
Repaint();
```

а для создания задержки (паузы) – функция

```
Sleep(t);
```

где t – время задержки в миллисекундах.

Очень сокращенная теория – см. л/р №9 из методички (часть 1).

Задание:

Написать *любую* программу, выводящую на экран какое-либо движущееся изображение, а также график какой-либо функции. На высокую оценку программа должна использовать все три вышеуказанных части теории и делать более-менее сложные движение либо рисование.

Упрощенный пример, как может выглядеть решение

(на невысокую оценку из-за примитивности рисунка и движения):

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
  for (int x=0; x<70;)          // x – координата для движения
  {
    Image1->Canvas->Pen->Color=clWhite;  // задание фонового цвета
    Image1->Canvas->Brush->Color=clWhite; // задание фонового цвета
    Image1->Canvas->Rectangle(x,10,x+30,40);  // стирание

    x+=3;         // перемещение вправо на 3 пикселя (движение)

    Image1->Canvas->Pen->Color=clRed; //задание цвета пера (контура)
    Image1->Canvas->Brush->Color=clGreen;
                                  // задание цвета кисти (заливки)
    Image1->Canvas->Ellipse(x,10,x+30,40); // рисование кружочка

    Repaint();  // обновление картинки на экране
    Sleep(80);  // пауза в 80 миллисекунд
  }
}
//--------------------------- Кнопка 2 ----------------------------------
// Внимание! В Chart предварительно должна быть создана линия (Series1).
// Для ее создания нужно дважды щелкнуть по Chart
void __fastcall TForm1::Button2Click(TObject *Sender)
{
  for (double X=-1; X<2; X+=0.01)
        Series1->AddXY(X, X*X);       // рисование параболы y=X*X
}
```

№9. Решение нелинейных уравнений:

Л/р №6 из методички (часть 2).

Цель работы – показать пример решения физико-математической задачи с помощью написания программы по заданному алгоритму.

Студент должен понять смысл понятия «корень функции», разницу между отделением и уточнением корней, понятие погрешности, а также используемый им алгоритм.

Студенты выполняют задание своего варианта. Однако в вариантах №1 и №6, где используется метод «простой» итерации, студенты при желании могут вместо него использовать любой другой метод (из-за

неопределенности способа выбора функции $\varphi(x)$ так, чтобы программа нашла все 3 корня).

№10. Побитовые операции

Необходимо на бумаге вычислить результат данного преподавателем выражения, содержащего побитовые операции над целочисленными операндами (возможно, также арифметические операции и операцию явного приведения типа), и при необходимости проверить его на компьютере.