Министерство науки и высшего образования Российской Федерации

Томский государственный университет
систем управления и радиоэлектроники

С.М. Алфёров

СОВРЕМЕННЫЕ СРЕДСТВА ПРОГРАММИРОВАНИЯ

Методические указания по самостоятельной и индивидуальной работе для студентов всех форм обучения и направлений магистратуры

Томск
2024

УДК 004.4
ББК 32.97
А-91

Рецензент:
Лукьянов А.К., доцент кафедры Автоматизированных систем управления ТУСУР, к.т.н.

Алфёров, Сергей Михайлович
А-91 Вычислительные системы: методические указания по самостоятельной и индивидуальной работе для студентов всех форм обучения и направлений магистратуры/ С.М. Алфёров – Томск: Томск. гос. ун-т систем упр. и радиоэлектроники, 2024. – 18 с.

Методические указания содержат задания к лабораторным работам, краткий теоретический материал для выполнения заданий и варианты.

Одобрено на заседании каф. Автоматизированных систем управления, протокол №11 от 23.11.2023.

УДК 004.4
ББК 32.97

Оглавление

1. Лабораторные работы

1.1. Потоки ввода и вывода.

Цель

Получить навыки создания и использования объектов ввода/вывода.

Общее задание

Изучить объектно-ориентированные средства ввода вывода: объекты cin, cout; классы fstream, ofstream, ifstream.

Объявить переменные: целочисленную, вещественную, символьную, строковую. Организовать ввод значений с клавиатуры при помощи cin. Вывести на экран значения при помощи cout двумя способами: простым и форматированным. Записать значения в два файла: текстовый и двоичный. Считать значения из обоих файлов, вывести на экран, убедиться в корректности совершенных операций.

Краткие теоретические сведения

Для вывода данных на экран или ввода с клавиатуры имеются объекты cout и cin, объявленные в файле **iostream.h**. Для работы с файлами имеются классы fstream, ofstream, ifstream, объявленные в файле **fstream.h**. Для форматированного вывода можно воспользоваться функциями setw, setprecision из заголовочного файла **iomanip.h**.

Важные рекомендации!

Для обеспечения чистоты эксперимента работы с файлами следует написать либо две программы (одна пишет, другая читает) либо в одной программе, но так, чтобы пользователь мог выбрать действие (чтение или запись), после выполнения этого действия программа должна закрыться.

Содержимое отчета

Титульный лист

Цель

Общее задание

Решение (Дракон-схема или UML-диаграмма деятельности алгоритма)

Текст программы

Результаты работы (скриншоты работы программы и содержимое файлов при разных входных данных и пояснения к ним).

Вывод

1.2. Инкапсуляция и агрегирование.

Цель

Научиться создавать классы с учетом правил предметной области. Уяснить необходимость скрытия некоторых свойств класса модификаторами private и protected.

Общее задание

Создать класс, имеющий заданные свойства, обеспечить доступ к свойствам (запись и чтение значений свойств) с учетом заданных ограничений. Все конструкторы класса так же должны обеспечивать создание объекта с учетом заданных ограничений. Написать программу, демонстрирующую сохранение ограничений при любых действиях внешней программы над объектом. Варианты заданий приведены в таблице 1.1.

Таблица 1.1 - Варианты заданий.

№	Класс	Свойства	Правила ПО
1	Монстр	Здоровье, сила	Суммарное значение здоровья и силы должно лежать в диапазоне от 2 до 150, каждое свойство должно лежать в диапазоне от 1 до 100. Если задаваемое значение не будет соответствовать условию, то автоматически присвоить ближайшее разрешенное.
2	Почтовый адрес	улица, дом, квартира	Первые символы до пробела свойства «улица» указывают на тип. «ул» - улица, «пр» - проспект, «пер» - переулок. На переулке номера домов лежат в диапазоне от 1 до 30, на улице от 1 до 100, на проспекте от 1 до 1000. Если дом частный, то квартира не указывается.
3	Результат Зачетной недели	Студент, семестр предметы, зачеты	Семестр может быть от 1 до 9. На 3, 7 семестрах студенты сдают по 3 зачета, на 5 и 6 по 4 зачета, на остальных по 5 зачетов. Написать функцию, определяющую можно ли студенту поставить штамп «зачтено», который ставится только при всех сданных зачетах.
4	Сессия ВУЗа (простое)	Семестр, Количество зачетов, количество экзаменов	Суммарное количество зачетов и экзаменов должно быть от 7 до 9.
5	Человек (сложное)	Голова, туловище, руки, ноги	Голова задается окружностью с заданными координатами центра и радиусом. Туловище задается прямоугольником: координатами верхнего левого угла, высотой и шириной. Руки и ноги задаются длинами и углами положения. Все члены не должны быть отделены друг от друга. Руки растут из верхних углов прямоугольника, ноги из нижних. Написать функцию определения роста человека.
6	Система прямоугольников	Прямоугольник1, Прямоугольник2	Прямоугольники задаются координатами. Площади прямоугольников не должны пересекаться и должны лежать в координатах от (0, 0) до (1000, 1000). Размеры прямоугольников не должны превышать 500.
7	Система фигур	Прямоугольник, окружность.	Прямоугольник задается координатами. Окружность задается координатами центра и радиусом. Площади фигур должны пересекаться. Написать функцию определения общей площади занимаемой фигурами.
8	Дата рождения пациента больницы	День, месяц, год	Год должен лежать в диапазоне от 1900 до 2014. Месяц может принимать значения от «январь» до «декабрь», день в зависимости от месяца и года (учитывая високосные года). Написать функцию для определения возраста

			пациента.
9	Отрезок	Точка 1, Точка2	Точки задаются координатами (x_1, y_1), (x_2, y_2) соответственно. При смене координат одной точки, координаты другой точки меняются по принципу: расстояние между точками не меняется, смещение другой точки минимально. Формулы для использования: $\varphi = \arctan\left(\frac{y_{12} - y_2}{x_{12} - x_2}\right)$ $x_{22}(\varphi) = L\cos(\varphi) + x_{12}$ $y_{22}(\varphi) = L\sin(\varphi) + y_{12}$ где x_{12}, y_{12} новые координаты точки1, x_{22}, y_{22} новые координаты точки2, L – длина отрезка. Аналогично при смене координат точка2.
10	Система отрезков на числовой прямой	Точка1, точка2, точка3. Расстояние 12, Расстояние 23.	Точки задаются числами. Сделать возможность определения и задания расстояния между точками. При измерении любого из расстояний, точка2 не меняется.
11	Файл	Имя, описатель	Сделать работу с файлом, подобно работе с массивом, т.е. возможность записи/чтения данных по индексу. Чтение/запись можно выполнять только в пределах индексов существующих элементов «массива». Можно добавлять один элемент в конец массива. Если индекс чтения/записи выходит за пределы существующих данных, то функция должна выдавать код ошибки.

1.3. Наследование.

Цель: познакомится с понятием наследования, научиться создавать иерархию классов.

Общее задание

Создать класс, имеющий заданные свойства, обеспечить доступ к свойствам (запись и чтение значений свойств) с учетом заданных ограничений. Все конструкторы класса так же должны обеспечивать создание объекта с учетом заданных ограничений. Написать программу, демонстрирующую сохранение ограничений при любых действиях внешней программы над объектом. Варианты заданий приведены в таблице 1.2.

Таблица 1.2 — Варианты заданий по теме «Наследование».

№	**Класс родительский/ дочерний**	**Свойства/ дополнительные свойства**	**Правила ПО**
1	Монстр	здоровье, сила	Суммарное значение здоровья и силы не должно превышать 150, каждое свойство должно лежать в диапазоне от 1 до 100. Если задаваемое значение не будет соответствовать условию, то

			автоматически присвоить ближайшее разрешенное.
	Демон	Разум	Разум лежит в диапазоне от 1 до 100. Суммарное значение всех свойств не должно превышать 200.
2	Почтовый адрес	Страна, область, город, улица, дом, квартира	Первые символы до пробела свойства «улица» указывают на тип. «ул» - улица, «пр» - проспект, «пер» - переулок. На переулке номера домов лежат в диапазоне от 1 до 30, на улице от 1 до 100, на проспекте от 1 до 1000. Если дом частный, то квартира не указывается.
	Галактический адрес	Расстояние от центра галактики, номер хвоста, название звездной системы, номер планеты, номер спутника при необходимости	Номер хвоста от 1 до 6 Номер планеты от 1 до 10 Номер спутника от 1 до 70
3	Результат Зачетной недели	Студент, семестр предметы, зачеты	Семестр может быть от 1 до 9. На 3, 7 семестрах студенты сдают по 3 зачета, на 5 и 6 по 4 зачета, на остальных по 5 зачетов. Написать функцию, определяющую можно ли студенту поставить штамп «зачтено», который ставится только при всех сданных зачетах.
	Дифференцированный зачет	Курсовые, оценки по курсовым	Оценку по курсовому можно ставить только при наличии зачета по соответствующей дисциплине
4	Сессия ВУЗа (простое)	Семестр, Количество зачетов, количество экзаменов	Суммарное количество зачетов и экзаменов не должно быть от 7 до 9.
	Весенняя сессия	Практика, оценка по практике	Значение оценки может быть: «н/а», «неуд», «удовл», «хор», «отл»
5	Человек	Голова, туловище, руки, ноги	Голова задается радиусом окружности. Туловище задается прямоугольником: высотой и шириной. Руки и ноги задаются длинами. Длины рук и ног должны быть в пределах от 110% до 130% от высоты туловища. Написать функцию определения роста человека.
	Киборг	Мощность экзоскелета	Не меньше площади прямоугольника туловища
6	Система прямоугольников	Прямоугольник1, Прямоугольник2	Прямоугольники задаются координатами. Площади прямоугольников не должны пересекаться и должны лежать в координатах от (0, 0) до (1000, 1000). Размеры прямоугольников не должны превышать 500.
		Прямоугольник3	

7	Система фигур	Прямоугольник, окружность.	Прямоугольник задается координатами. Окружность задается координатами центра и радиусом. Площади фигур должны пересекаться. Написать функцию определения общей площади занимаемой фигурами.
	Система из трех фигур	Окружность2.	Каждая из фигур должна пересекаться хотя-бы с одной из других. Написать функцию определения общей площади занимаемой фигурами. Для этого воспользоваться методом Монте-Карло.
8	Дата рождения пациента больницы	День, месяц, год	Год должен лежать в диапазоне от 1900 до 2018. Месяц может принимать значения от «январь» до «декабрь», день в зависимости от месяца и года (учитывая високосные года). Написать функцию для определения возраста пациента.
	Пациент	ФИО	Строка, содержащая ФИО должна содержать хотя-бы один пробел.
9	Отрезок	Точка 1, Точка2	Точки задаются координатами (x_1, y_1), (x_2, y_2) соответственно. При смене координат одной точки, координаты другой точки меняются по принципу: расстояние между точками не меняется, смещение другой точки минимально. Формулы для использования: $\varphi = \arctan\left(\frac{y_{12} - y_2}{x_{12} - x_2}\right)$ $x_{22}(\varphi) = L\cos(\varphi) + x_{12}$ $y_{22}(\varphi) = L\sin(\varphi) + y_{12}$ где x_{12}, y_{12} новые координаты точки1, x_{22}, y_{22} новые координаты точки2, L – длина отрезка. Аналогично при смене координат точка2.
	Ломаная	Точка 3	
10	Табурет	Высота, количество ножек, материал	Высота от 20 до 70 см. количество ножек: 1,3,4 или 5 материал: пластмасса, дерево или металл
	Стул	Высота спинки	От 80% до 150% от высоты
11	Файл-массив	Имя, описатель (типа int или FILE), количество элементов	Сделать работу с файлом, подобно работе с массивом, т.е. возможность записи/чтения данных по индексу. Если индекс чтения/записи выходит за пределы существующих данных, то функция должна выдавать код ошибки.
	Файл-матрица	Количество строк	Сделать работу с файлом, подобно

			работе с матрицей, т.е. возможность записи/чтения данных по индексам. Если индексы чтения/записи выходят за пределы существующих данных, то функция должна выдавать код ошибки.

1.4. Модификация имеющихся классов.

Цель

Получить навык использования имеющихся классов для создания новых классов.

Общее задание

Взять готовый класс и создать на основе его класс-наследник с дополнительными возможностями согласно варианту.

Вариант 1

Сделать простой аналог Combobox из TextBox, Button, ListBox

Вариант 2

Сделать на основе Combobox класс advCombobox с возможностью фильтрации значений в списке по введенной строке и выбор из отфильтрованных значений.

Вариант 3

Сделать поле ввода текста по маске, маска должна задаваться в конструкторе при создании. Маска задается элементами: «9» — цифра, «a» – латинская буква, «-» символ минус. Например, если маска 999, то допустимые значения: как-бы числа от 000 до 999.

Вариант 4

Сделать поле ввода текста, с допустимыми значениями, заданными массивом строк, который задается в конструкторе при создании.

Вариант 5

Сделать перемещаемую картинку (при нажатии мышкой на картинку и удержании, можно её перемещать по форме).

Вариант 6

Сделать «живую» кнопку (при нахождении указателя мышки над кнопкой, кнопка окрашивается в яркий цвет, при покидании, кнопка возвращает свой цвет обратно).

Содержание отчета

1. Титульный.
2. Цель работы.
3. Индивидуальное задание.
4. Решение (UML-диаграмма классов и текст программы).
5. Результаты работы.
6. Вывод.

Пример, анимация по движению мышки над объектом на основе картинки.

unit2.h

```
#ifndef Unit2H
#define Unit2H
#include <ExtCtrls.hpp>
#include <Graphics.hpp>
class me_image : public TImage{
  Graphics::TBitmap **arrBMP;
  int n,nmax,index;
public:
  __fastcall virtual me_image(Classes::TComponent* AOwner);
  __fastcall virtual ~me_image();
```

```
  int addPicture(const char* filename);
  bool setCurBMP(int index);
  DYNAMIC void __fastcall MouseMove(Classes::TShiftState Shift, int X, int Y);
  int getN();
};
#endif
```

unit2.cpp

```
#pragma hdrstop
#include "Unit2.h"
__fastcall me_image::me_image(Classes::TComponent* AOwner):TImage(AOwner){
  n=0;nmax=60; index=-1;
  arrBMP=new Graphics::TBitmap*[nmax];
  for (int i=0; i< nmax; i++) {  arrBMP[i]=NULL;  }
}
__fastcall me_image::~me_image(){
  this->Picture->Bitmap=NULL;
  for (int i=0; i< n; i++) { delete arrBMP[i];  }
  delete [] arrBMP;
}
int me_image::addPicture(const char* filename){
          if (n<nmax) {
           arrBMP[n]=new Graphics::TBitmap;
           arrBMP[n]->LoadFromFile(filename);
           if (n==0) {
             index=0;
             this->Width=arrBMP[0]->Width;
             this->Height=arrBMP[0]->Height;
           }
           n++;
           return n;
          }
          return 0;
}
bool me_image::setCurBMP(int index){
          if (index>=0 && index<n) {
                    this->index=index;
                    this->Picture->Bitmap=arrBMP[index];
                    return true;
          }
          return false;
}
void __fastcall me_image::MouseMove(Classes::TShiftState Shift, int X, int Y){
  TImage::MouseMove(Shift, X, Y);
  index=(index +1)%n;
  this->Picture->Bitmap=arrBMP[index];
}
int me_image::getN(){  return n; }
#pragma package(smart_init)
```

unit1.h

```
#ifndef Unit1H
#define Unit1H
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <ExtCtrls.hpp>
#include <Unit2.h>
#include <string.h>
#include <stdlib.h>
class TForm1 : public TForm{
__published:   // IDE-managed Components
        TImage *Image1;
        TButton *Button1;
        TButton *Button2;
        TButton *Button3;
        void __fastcall FormCreate(TObject *Sender);
        void __fastcall Button1Click(TObject *Sender);
        void __fastcall Button2Click(TObject *Sender);
        void __fastcall Button3Click(TObject *Sender);
private:        // User declarations
        me_image *hyp_img, *super_img;
        int index;
public:         // User declarations
        __fastcall TForm1(TComponent* Owner);
};
extern PACKAGE TForm1 *Form1;
#endif
```

unit1.cpp

```
#include <vcl.h>
#pragma hdrstop
#include "Unit1.h"
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
__fastcall TForm1::TForm1(TComponent* Owner): TForm(Owner){
 int old_cnt=this->ControlCount;
 index=0;
 hyp_img=new me_image(this);
 hyp_img->Visible=true;
 this->InsertControl(hyp_img);
 if (old_cnt+1==this->ControlCount) {
   MessageBox(0,"add cntrl - ok!!!","msg",0);
 }
 hyp_img->addPicture("..\\anim1\\subzero1.bmp");
 hyp_img->addPicture("..\\anim1\\subzero1.bmp");
 hyp_img->addPicture("..\\anim1\\subzero2.bmp");
 hyp_img->addPicture("..\\anim1\\subzero2.bmp");
```

```
hyp_img->addPicture("..\\anim1\\subzero3.bmp");
hyp_img->addPicture("..\\anim1\\subzero3.bmp");
hyp_img->addPicture("..\\anim1\\subzero4.bmp");
hyp_img->addPicture("..\\anim1\\subzero4.bmp");
hyp_img->addPicture("..\\anim1\\subzero5.bmp");
hyp_img->addPicture("..\\anim1\\subzero5.bmp");
hyp_img->addPicture("..\\anim1\\subzero6.bmp");
hyp_img->addPicture("..\\anim1\\subzero6.bmp");
hyp_img->addPicture("..\\anim1\\subzero7.bmp");
hyp_img->addPicture("..\\anim1\\subzero7.bmp");
hyp_img->addPicture("..\\anim1\\subzero6.bmp");
hyp_img->addPicture("..\\anim1\\subzero6.bmp");
hyp_img->addPicture("..\\anim1\\subzero5.bmp");
hyp_img->addPicture("..\\anim1\\subzero5.bmp");
hyp_img->addPicture("..\\anim1\\subzero4.bmp");
hyp_img->addPicture("..\\anim1\\subzero4.bmp");
hyp_img->addPicture("..\\anim1\\subzero3.bmp");
hyp_img->addPicture("..\\anim1\\subzero3.bmp");
hyp_img->addPicture("..\\anim1\\subzero2.bmp");
hyp_img->addPicture("..\\anim1\\subzero2.bmp");
hyp_img->Left=300;
hyp_img->Top=300;
super_img=new me_image(this);
super_img->Visible=true;
this->InsertControl(super_img);
super_img->Left=0;
super_img->Top=0;
super_img->addPicture("..\\anim2\\char1.bmp");
super_img->addPicture("..\\anim2\\char2.bmp");
super_img->addPicture("..\\anim2\\char3.bmp");
super_img->addPicture("..\\anim2\\char4.bmp");
super_img->addPicture("..\\anim2\\char5.bmp");
super_img->addPicture("..\\anim2\\char6.bmp");
super_img->addPicture("..\\anim2\\char7.bmp");
super_img->addPicture("..\\anim2\\char8.bmp");
super_img->addPicture("..\\anim2\\char9.bmp");
super_img->addPicture("..\\anim2\\char10.bmp");
super_img->addPicture("..\\anim2\\char11.bmp");
super_img->addPicture("..\\anim2\\char12.bmp");
super_img->addPicture("..\\anim2\\char13.bmp");
super_img->addPicture("..\\anim2\\char14.bmp");
super_img->addPicture("..\\anim2\\char15.bmp");
super_img->addPicture("..\\anim2\\char16.bmp");
super_img->addPicture("..\\anim2\\char15.bmp");
super_img->addPicture("..\\anim2\\char14.bmp");
super_img->addPicture("..\\anim2\\char13.bmp");
super_img->addPicture("..\\anim2\\char12.bmp");
super_img->addPicture("..\\anim2\\char11.bmp");
super_img->addPicture("..\\anim2\\char10.bmp");
super_img->addPicture("..\\anim2\\char9.bmp");
```

```
 super_img->addPicture("..\\anim2\\char8.bmp");
 super_img->addPicture("..\\anim2\\char7.bmp");
 super_img->addPicture("..\\anim2\\char6.bmp");
 super_img->addPicture("..\\anim2\\char5.bmp");
 super_img->addPicture("..\\anim2\\char4.bmp");
 super_img->addPicture("..\\anim2\\char3.bmp");
 super_img->addPicture("..\\anim2\\char2.bmp");
 super_img->Show();
}
void __fastcall TForm1::FormCreate(TObject *Sender){
 Image1->Picture->Bitmap->LoadFromFile("pic.bmp");
 Image1->Width=Image1->Picture->Bitmap->Width;
 Image1->Height=Image1->Picture->Bitmap->Height;
}
void __fastcall TForm1::Button1Click(TObject *Sender){
 index=(index+1) % 10;
 this->hyp_img->setCurBMP(index);
 this->hyp_img->Refresh();
 char s[10];
 itoa(index,s,10);
 this->Caption=s;
}
void __fastcall TForm1::Button2Click(TObject *Sender) {
 if (index>0) index--;
 this->hyp_img->setCurBMP(index);
 this->hyp_img->Refresh();
 char s[10];
 itoa(index,s,10);
 this->Caption=s;
}
void __fastcall TForm1::Button3Click(TObject *Sender) {
 char s[10];
 itoa(hyp_img->getN(),s,10);
 this→Caption=s;
}
```

1.5. Организация вычислений в параллельных потоках.

Цель

Получить навыки распараллеливания задач и написания параллельных потоков для неразделяемых ресурсов.

Общее задание

Дана прямоугольная матрица заданного типа. “Разделить” матрицу на подматрицы произвольным способом. Каждый поток должен обрабатывать отдельную подматрицу так, чтобы в целом решалась задача, соответствующая варианту. Искомое значение должно быть оформлено как общий ресурс для всех потоков, таким образом он будет не разделяемым. Разрешить проблему неразделяемого ресурса при помощи симафоров или мютексов.

Проверить правильность решенной задачи вычислив искомое значение в основном потоке. Запустить программу при разном количестве столбцов и строк матрицы.

Построить графики зависимости времени обработки матрицы от количества элементов матрицы (сложности задачи). Выявить сложность задачи, при которой параллельный способ обрабатывает матрицу быстрее последовательного.

Вариант 1

Тип матрицы: целочисленная.

Искомое значение: количество элементов, равных нулю, сумма индексов которых, является четным числом.

Вариант 2

Тип матрицы: вещественная.

Искомое значение: количество элементов входящих в заданный диапазон.

Вариант 3

Тип матрицы: вещественная. Матрица содержит координаты точек на плоскости. В четных столбцах записаны координаты X точек. В нечетных, координаты Y.

Искомое значение: количество точек, находящихся дальше заданного расстояния от начала координат.

Вариант 4

Тип матрицы: вещественная.

Искомое значение: сумма элементов входящих в заданный диапазон.

Вариант 5

Тип матрицы: целочисленная, однобайтные.

Искомое значение: сумма номеров столбцов, элементы которых равны нулю.

Вариант 6

Тип матрицы: целочисленная, однобайтные элементы.

Искомое значение: количество последовательностей элементов 1, 2, 3 в строках матрицы.

Вариант 7

Тип матрицы: вещественная.

Искомое значение: количество элементов удовлетворяющих условию положительный элемент, стоящий в нечетном столбце или элемент со значением меньше -10, стоящий в четном столбце.

Вариант 8

Тип матрицы: целочисленная, однобайтные элементы.

Искомое значение: количество соседних в строке пар элементов, равных по значению и противоположных по знаку.

Вариант 9

Тип матрицы: целочисленная, однобайтные элементы.

Искомое значение: количество элементов, после которых, в строке через одну позицию, находится точно такое же число. То есть, количество элементов $A_{i,\ j}$, удовлетворяющих условию $A_{i,j} = A_{i,j+2}$.

Содержание отчета:

1. Титульный
2. Цель работы
3. Задание
   3.1. Общее
   3.2. Вариант
4. Решение задачи
   4.1. UML-диаграмма деятельности
   4.2. Код программы
5. Результаты
6. Вывод

1.6. Абстрактные классы.

Цель

Получить навыки создания иерархии классов с использованием абстрактных классов; уяснить пользу от использования абстрактных классов.

Общее задание

Создать указанный абстрактный класс. Создать несколько производных классов, реализующие поведение абстрактного класса. Продемонстрировать возможность работы с объектами разных классов через методы абстрактного класса. Варианты заданий приведены в таблице 1.3.

Таблица 1.3 — Варианты заданий для темы «Абстрактные классы».

Абстрактный класс (интерфейс)	**Дочерние/производные классы (драйверы)**
1а) Мышь Вход: *x,y* – начальные *s* – чувствительность Выход *x,y* – текущие	1) Шариковая мышь Координаты определяются по потоку байтов (8 бит), младшая часть 4 бит – положение X-валика старшая часть 4 бит – положение Y-валика 2) Оптическая мышь, Координаты определяются по потоку кадров (64 бит) Матрица битов 8x8 – монохромный кадр рельефа поверхности
1б) Мышь Вход: *x,y* – начальные *s* – чувствительность Выход *x,y* – текущие	1) Шариковая мышь Координаты определяются по потоку байтов, 1-й байт – положение X-валика 2-й байт – положение Y-валика 2) Оптическая мышь, Координаты определяются по потоку кадров (64 байта) Матрица 4x4 байт – монохромный кадр рельефа поверхности
2) Двигатель Вход: *f* – требуемая частота вращения двигателя Выход: драйвер должен сформировать управляющие сигналы	1. Шаговый Для вращения с определенной частотой необходимо передавать в порт последовательность байтов с соответствующим временным шагом (4 старших бита не используются): 00000001b, 00000010b, 00000100b, 00001000b Для вращения в обратном направлении байты передавать в обратном порядке. Чтобы симитировать выдачу сигналов с определенным временным шагом, вместе с байтом можно отобразить временную метку. 2. Постоянного тока Для вращения с заданной частотой необходимо однократно передать в контроллер последовательность: код символа 'u' (байт), значение напряжения в В (float). Для подтверждения задания частоты принять от контроллера сообщение: код символа 'f' (байт), значение частоты вращения в Гц (float). Для вращения в обратном направлении передавать значение напряжения с противоположным знаком.
3) Задатчик давления Вход: *P* – требуемое давление Выход: драйвер должен сформировать управляющие	1. Клапанный Давление в системе формируется по формуле $P = \frac{U_1 \cdot P_{in} + U_2 \cdot P_{out}}{U_1 + U_2}$ U_1, U_2 – управляющие напряжения на клапанах, P_{in}, P_{out} – давление на входе и выходе системы 2. Прессовый Давление в системе формируется по формуле:

сигналы	$P = \frac{(P_0 + 1\ atm) \cdot V_0}{V} - 1\ atm$ $V = V_0 - k \sum_{i=1}^{n} (f_i \cdot dt_i)$ P_0 – начальное давление, V_0, V – начальный и текущий объем воздуха в прессе, k – коэффициент пропорциональности, f_i – последовательность управляющих сигналов, dt_i – длительности управляющих сигналов
4) Стек	1. Стек на основе статического массива 2. Стек на основе динамического массива
5) Очередь	1. Очередь на основе статического массива 2. Очередь на основе динамического массива
6) Принтер Вход: *C* – Печатаемый символ, *F* – шрифт. Драйвер должен выдать изображение	Матричный, способен печатать только монохромное изображение Для печати изображения должен выдать матрицу битов 8x8 Струйный, способен печатать цветное изображение Для печати изображения должен выдать матрицу байтов 8x8
7) Накопитель Вход: Запись информации *D* в блок *N*	1) Дисковый Блок N преобразуется в сторону *S*, дорожку *T*, сектор *C*, по формуле: $N = CT \cdot CC \cdot S + CC \cdot T + C$ Тогда $C = N \bmod CC$ $T = (N \div CC) \bmod CT$ Или $T = ((N - C) \div CC) \bmod CT$ $S = N \div (CC \cdot CT)$ или $S = ((N - C) \div CC - T) \div CT$ *CC* – кол-во секторов на дорожке, *CT* – кол-во дорожек на стороне. Запись информации производится на сторону *S*, дорожку *T*, сектор *C*. 2) Flash Запись производится в ячейки начиная с адреса *N*·512.

Содержание отчета:

1. Титульный
2. Цель работы
3. Задание
3.1. Общее
3.2. Вариант
4. Решение задачи
4.1. UML-диаграмма классов
4.2. Код программы
5. Результаты
6. Вывод

1.7. Решение задач в Scilab.

Цель

Получить навыки программирования в среде Scilab.

Общее задание

Написать программу для решения системы нелинейных уравнений.

Алгоритм решения задачи:

1. Выбрать начальное приближение.

2. Цикл до момента, когда расстояние между соседними приближения окажется меньше требуемой точности

2.1. Сформировать якобиан (матрицу частных производных).

2.2. По якобиану и текущему приближению определить следующее приближение.

Варианты заданий

1. $\begin{cases} \sin(x-1) = 1{,}3 - y \\ x - \sin(y+1) = 0{,}8 \end{cases}$

2. $\begin{cases} \sin y + 2x = 2 \\ \cos(x-1) + y = 0{,}7 \end{cases}$

3. $\begin{cases} \cos(x-1) + y = 0{,}5 \\ x - \cos y = 3 \end{cases}$

4. $\begin{cases} \cos y + x = 1{,}5 \\ 2y - \sin(x-0{,}5) = 1 \end{cases}$

5. $\begin{cases} \sin(x+1) - y = 1{,}2 \\ 2x + \cos y = 2 \end{cases}$

6. $\begin{cases} \sin(y+0{,}5) - x = 1 \\ \cos(x-2) + y = 0 \end{cases}$

7. $\begin{cases} \sin x + 2y = 2 \\ \cos(y-1) + x = 0{,}72 \end{cases}$

8. $\begin{cases} \cos(y+0{,}5) + x = 0{,}8 \\ \sin x - 2y = 1{,}6 \end{cases}$

9. $\begin{cases} \cos x + y = 1{,}5 \\ 2x - \sin(y-0{,}5) = 1 \end{cases}$

10. $\begin{cases} \sin(y-1) + x = 1{,}3 \\ y - \sin(x+1) = 0{,}8 \end{cases}$

11. $\begin{cases} \sin(x+0{,}5) - y = 1 \\ \cos(y-2) + x = 0 \end{cases}$

12. $\begin{cases} 2x - \cos(y+1) = 0 \\ y + \sin x = -0{,}4 \end{cases}$

13. $\begin{cases} 2y - \cos(x+1) = 0 \\ x + \sin y = -0{,}4 \end{cases}$

14. $\begin{cases} \cos(y+0{,}5) - x = 2 \\ \sin x - 2y = 1 \end{cases}$

15. $\begin{cases} \cos(x+0{,}5) - y = 2 \\ \sin y - 2x = 1 \end{cases}$

16. $\begin{cases} \sin(x+1) - y = 1 \\ 2x + \cos y = 2 \end{cases}$

Содержание отчета

Аналогично лабораторной 1.5.

2. Список рекомендуемой литературы

2.1. Основная литература

1. Казанский, А. А. Программирование на Visual C# : учебное пособие для вузов / А. А. Казанский. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2022. — 192 с. [Электронный ресурс]: — Режим доступа: https://urait.ru/bcode/470261 (дата обращения: 22.11.2023).

2. Капитанов, Д. В. Введение в SciLab : учебное пособие / Д. В. Капитанов, О. В. Капитанова. — Нижний Новгород : ННГУ им. Н. И. Лобачевского, 2019. — 56 с. [Электронный ресурс]: — Режим доступа: https://e.lanbook.com/book/144676 (дата обращения: 22.11.2023).

2.2. Дополнительная литература

1. Параллельное программирование на языке C# : учебно-методическое пособие / составитель Р. Х. Ахмадулин. — Тюмень : ТюмГНГУ, 2016. — 37 с. [Электронный ресурс]: — Режим доступа: https://e.lanbook.com/book/88569 (дата обращения: 22.11.2023).

2.3. Перечень пособий, методических указаний и материалов, используемых в учебном процессе

1. Боровской, И.Г. Специализированная подготовка разработчиков бизнес приложений [Электронный ресурс]: учебное пособие / И. Г. Боровской, С. И. Колесникова, А. А. Матолыгин; Томский государственный университет систем управления и радиоэлектроники (Томск). - Электрон. текстовые дан. - Томск: [б. и.], 2012. - on-line, 256 с. [Электронный ресурс]: — Режим доступа: http://edu.tusur.ru/training/publications/2532 (дата обращения: 22.11.2023).