МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Воронежский государственный технический университет»

Кафедра кадастра недвижимости, землеустройства и геодезии

Информационное обеспечение задач космической геодезии и навигации

МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ

к выполнению лабораторных и практических работ для студентов
направления 21.04.03 «Геодезия и дистанционное зондирование»
(программа «Инженерная геодезия») всех форм обучения

Воронеж 2022

1. ПОСТАНОВКА ЗАДАЧИ

Задача: произвести уравнивание превышений и вычисление высот реперов по результатам нивелирования III класса. Произвести оценку точности результатов полевых измерений и уравненных значений [1].

При проложении нивелирных ходов и сетей III класса производятся не только необходимые измерения для вычисления отметок реперов, но и избыточные измерения для проведения контроля измерений, повышения точности окончательных результатов, оценки точности полевых измерений и полученных окончательных результатов. Наличие избыточных измерений позволяет производить уравнивание измеренных величин, целью которого является устранение невязок, возникающих в данном ряде измерений, путем вычисления и введения поправок в измеренные величины.

В одиночном нивелирном ходе полученная невязка с обратным знаком распределяется между превышениями по отдельным секциям хода пропорционально их длинам (рис. 1).

По полученным уравненным превышениям вычисляют высоты определяемых реперов. Оценка точности полевых измерений в одиночном ходе производится по разностям двойных измерений.

Контрольные вопросы:

а) позволяет ли уравнивание найти наиболее надежные результаты?
б) зачем нужны избыточные измерения?

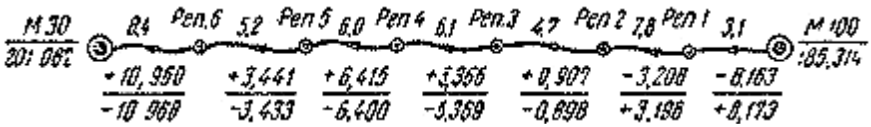


Рис. 1. Схема одиночного нивелирного хода

2. МАТЕМАТИЧЕСКОЕ ОБОСНОВАНИЕ РЕШЕНИЯ ЗАДАЧИ

Для вычислительной обработки полевых измерений тщательно проверяют их результаты в полевых журналах.

По данным нивелирования составляют схему хода, на которую выписывают исходные данные (номера и высоты исходных реперов, номера определяемых реперов и номера секций) и измеренные величины (длины секций, измеренные превышения), а также указывают стрелками направление нивелирования, как по каждой секции, так и по всему ходу.

Вычислительная обработка результатов нивелирования состоит из двух этапов — оценки качества полученных результатов и уравнительных вычислений для окончательного определения высот реперов.

Вычисления проводят в специальных формулярах (таблицах), облегчающих процесс вычислений и наглядно представляющих результаты измерений, вычислений и их контроля, полученных окончательных результатов.

Оценка качества нивелирования

Оценка качества нивелирования заключается в сравнении полученных расхождений превышений по секциям из прямого и обратного хода с допустимыми величинами, а также полученной невязки в превышениях с допустимой невязкой хода, для данного класса нивелирования.

Если расхождения превышают допустимые значения, то нивелирование повторяется.

Для вычислительной обработки в графы 1–5 и 13 (табл. 1) выписывают исходные данные и измеренные величины.

Далее вычисляют расхождения в превышениях по секциям из прямого и обратного хода по формуле

$$d = h_{\text{пр}} + h_{\text{обр}}, \tag{1}$$

где $h_{\text{пр}}$ — превышение из прямого хода; $h_{\text{обр}}$ — превышение из обратного хода.

Вычисленные расхождения d контролируют суммированием величин, записанных в графах 4,5 по формуле

$$\sum h_{\text{пр}} + \sum h_{\text{обр}} = \sum d \tag{2}$$

и сравнением полученной суммарной разности $\sum d$ с суммой расхождений превышений по секциям из прямого и обратного хода (табл. 2 графа 7). Сходимость полученных сумм должна быть полной.

Таблица 1

Часть ведомости с исходными данными и измеренными величинами

№ секций хода	№№ марок и реперов	Длины секций, L_i	Превышения		……	Высота, H_i
			Прямой ход, h_i	Обратный ход, h_i		
1	2	3	4	5	……	13
	100					185,314
1		3,1	–8,163	+8,173		
	11					
2		7,8	–3,208	+3,196		
	12					
3		4,7	+0,907	–0,898		
	13					
4		6,1	+5,355	–5,369		
	14					
5		6,0	+6,415	–6,400		
	15					
6		5,2	+3,441	–3,433		
	16					
7		8,4	+10,950	–10,968		
	30					201,062
		41,3	+15,697	–15,699	……	

Таблица 2

Часть ведомости для размещения расчетных данных

№ п/п	Среднее h_i, мм	Расхождения		d_i^2	$\frac{d_i^2}{L_i}$	Поправка V_i, мм	Исправленные превышения $h+v_i$, м	Высота, м	Вес отметок	Средняя квадратическая погрешность	Средняя квадратическая погрешность
		Полученные d_i, мм	Предельно-допустимые, мм								
1	6	7	8	9	10	11	12	13	14	15	16

Полученные расхождения в превышениях по секциям из прямого и обратного хода сравнивают с допустимыми значениями приведенными в инструкции для данного класса нивелирования. Для Ш класса нивелирования расхождения в превышениях по секциям из прямого и обратного хода не должны превышать значений, вычисленных по формуле

$$f_{h_{\text{пред}}} = 10\text{мм}\sqrt{L_i} \tag{3}$$

где L — длина секции в км.

Для вычисления невязки в превышениях хода в графе 6 (см. табл. 2) вычисляют средние из абсолютных значений прямого и обратного превышений. У полученных величин проставляют знак, соответствующий прямому направлению хода. Вычисление средних превышений контролируют выражениями:

$$\frac{\sum\left|h_{\text{пр}}\right| + \left|h_{\text{обр}}\right|}{2} = \sum\left|h_{\text{ср}}\right| \tag{4}$$

Вычисляют невязку в превышениях по ходу:

$$f_h = \sum h_{\text{ср}} - (H_{\text{к}} - H_{\text{н}}), \tag{5}$$

где $\sum h_{\text{ср}}$ — сумма средних значений прямых и обратных превышений по всему ходу; $H_{\text{к}}$, $H_{\text{н}}$ — отметки исходных начального и конечного реперов.

Полученное значение невязки сравнивают с допустимым значением для данного класса нивелирования по формуле (3).

В случае, когда значение полученной невязки превышает допустимое, то тщательно проверяют записи в полевых журналах и результаты последующих вычислений. Если ошибок не обнаружено, то нивелирование повторяют.

Уравнительные вычисления

Далее выполняют уравнивание превышений и вычисление окончательных высот определяемых реперов. Для этого в графе 11 (см. табл. 2) поправки в измеренные превышения по каждой секции вычисляют по формуле

$$v_{h_i} = -\frac{f_h}{L} L_i \tag{6}$$

Контролируют вычисление поправок суммированием их по секциям по формуле:

$$[v_{h_i}] = -f_h \tag{7}$$

В графе 12 вычисляют исправленные превышения по формуле

$$h_{\text{испр}} = h_{\text{изм}} + v_h \tag{8}$$

с контролем вычислений исправленных превышений по формуле

$$\sum h_{\text{испр}} = H_{\text{кон}} - H_{\text{нач}} \tag{9}$$

Высоты определяемых реперов (графа 13) вычисляют по формуле

$$H_{i+1} = H_i + h_{\text{испр}}, \tag{10}$$

где H_{i+1} — высота последующей точки; H_i — высота предыдущего репера.

Заключительным контролем является абсолютное схождение отметок исходного и конечного репера

$$H_{\text{выч.кон}} = H_{\text{исх.кон}}. \tag{11}$$

Математическое обоснование играет решающую роль в успешной разработке приложения.

Контрольные вопросы:

а) как определяется невязка по ходу?

б) какими выражениями контролируют вычисление средних превышений?

в) как производится оценка качества выполненного нивелирования?

Программирование на языке С++ подробно рассмотрено в [2–14].

3. СОЗДАНИЕ ПРИЛОЖЕНИЯ В СРЕДЕ VISUAL C++

3.1. Создание каркаса приложения

При разработке приложения использованы классы MFC Library: CDialog, CEdit, CButton, CMSFlexGrid, CPicture, CColorStatic, COleFont.

Запускаем Visual C++ и получаем главное меню (рис. 2), в котором из пункта File главного меню надо выбрать New (клик левой кнопкой).

При этом появится диалоговая панель New (рис. 3), в которой надо выбрать MFC APPWizard (exe), а в окне Project name надо ввести название проекта. В окне Location можно выбрать место, где будет храниться проект. По завершении надо нажать внизу на кнопку OK.

В диалоговой панели MFC AppWizard-Step 1 (рис. 4) надо щелкнуть радиокнопку Dialog based и нажать кнопку Next

В диалоговой панели MFC AppWizard-Step 2 (рис. 5) оставить все без изменений и нажать Next

В диалоговой панели MFC AppWizard-Step 3 (рис. 6) оставить все без изменений и нажать Next

В диалоговой панели MFC AppWizard-Step 4 (рис. 7) оставить все без изменений и нажать Finish

Появляется диалоговая панель New Project Information (рис. 8), в которой сведены вместе все спецификации проекта. Надо ознакомиться и нажать OK.

На полученном каркасе приложения (рис. 9) убираем кнопку «Cancel» и надпись «Todo: Place dialog controls here»

Получаем диалоговую панель (рис. 10).

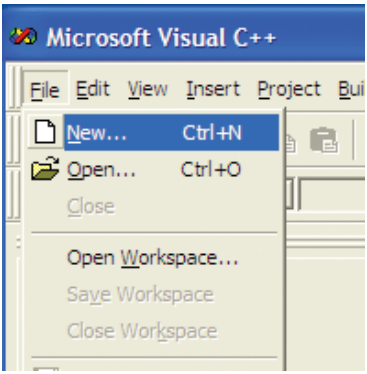


Рис. 2. Главное меню Visual C++

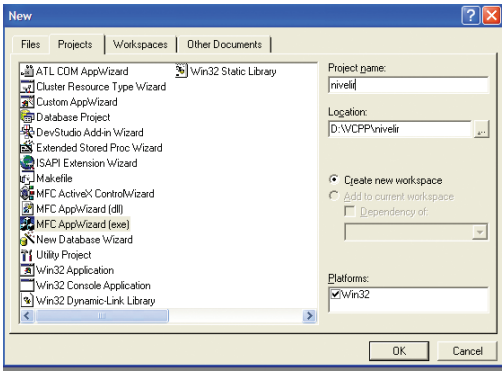


Рис. 3. Диалоговая панель New

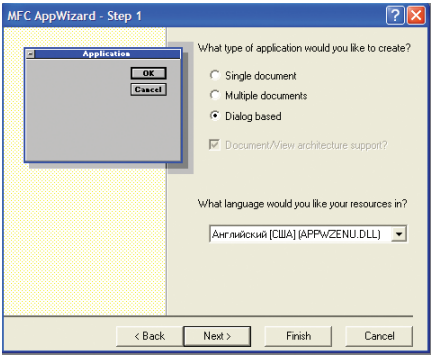


Рис. 4. Диалоговая панель MFC AppWizard-Step 1

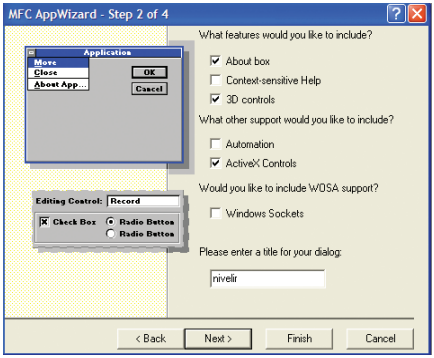


Рис. 5. Диалоговая панель MFC AppWizard-Step 2

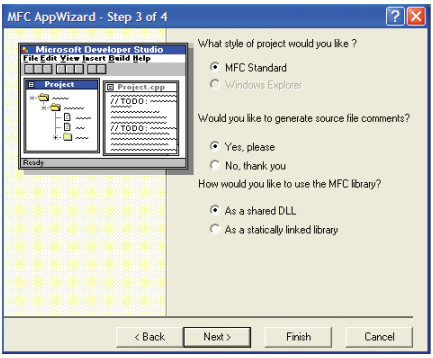


Рис. 6. Диалоговая панель MFC AppWizard-Step 3

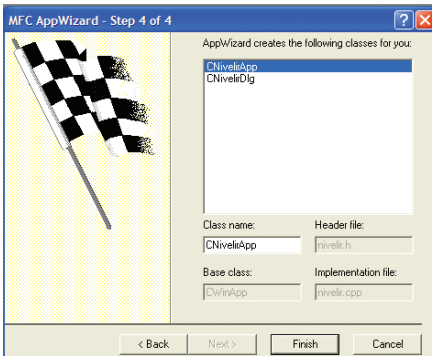


Рис. 7. Диалоговая панель MFC AppWizard-Step 4

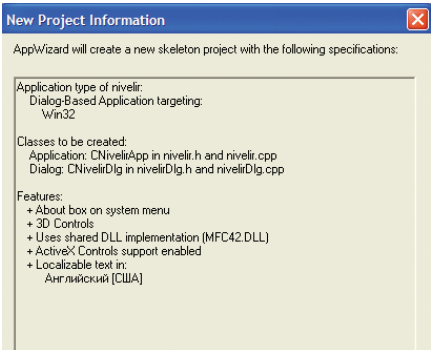


Рис. 8. панель New Project Information

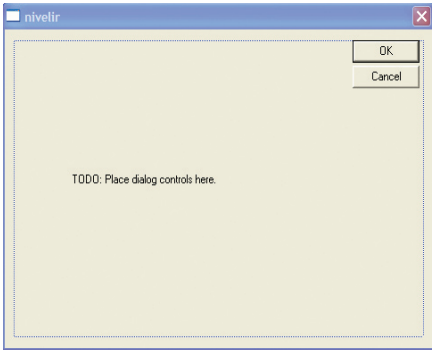


Рис. 9. Полученный каркас приложения

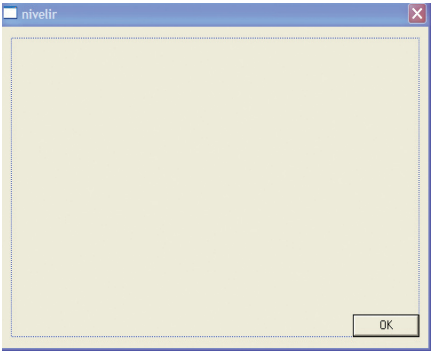


Рис. 10. Диалоговая панель приложения

3.2. Размещение рисунка на диалоговой панели

Разместим рисунок одиночного нивелирного хода на диалоговой панели. Для этого необходимо проделать ряд операция(манипуляций). В главном меню (рис. 11) выбираем Insert, затем Resource.

На панели Insert Resource выбираем Bitmap и нажимаем кнопку Import (рис. 12).

На панели Import Resource открыть папку Res и установить тип файлов «Все файлы (*.*» (рис. 13).

В папке Res выбрать название изображения (оно должно быть в формате *.bmp) и нажать Import (рис. 14).

На панели Controls выбрать Picture (рис. 15) и расположить на диалоговой панели разработки:

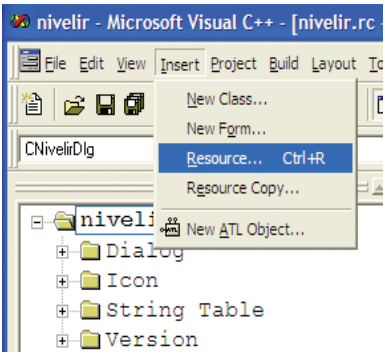


Рис. 11. Раздел Insert главного меню

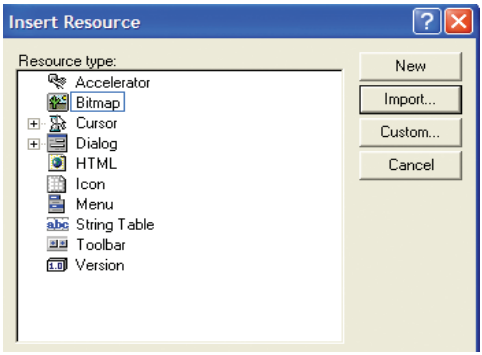


Рис. 12. Панель Insert Resource

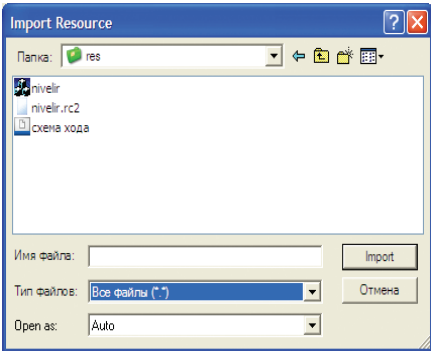


Рис. 13. Панель Import Resource

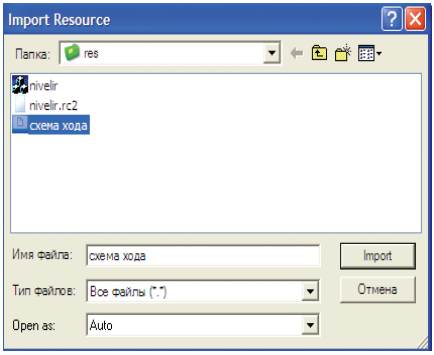


Рис. 14. Панель Import Resource

Выделить изображение и открыть панель свойств (рис. 16).

В панели свойств изображения Picture Properties выбрать в списке Bitmap и IDB_BITMAP1 (рис. 17).

В результате получаем на диалоговой панели изображение нивелирного хода (рис. 18).

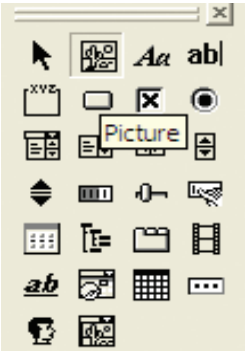


Рис. 15. Панель Controls

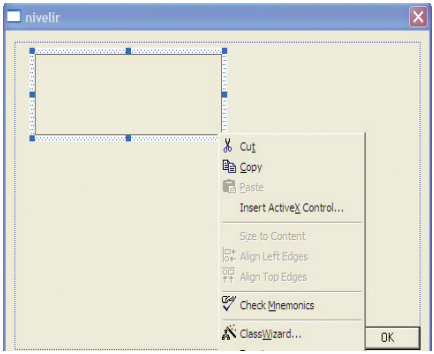


Рис. 16. Диалоговая панель с размещенным на ней контуром изображения

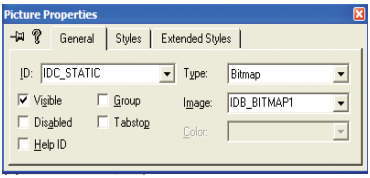


Рис. 17. Панель свойств изображения

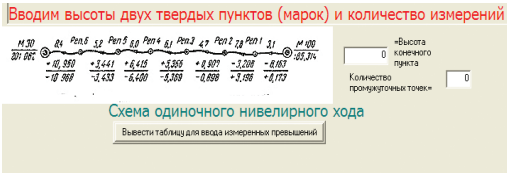


Рис. 18. Изображение схемы нивелирного хода в формате *.bmp

3.3. Создание окон редактирования

Для ввода исходных данных создадим окна редактирования. Для этого надо на панели Controls выбрать соответствующую пиктограмму (рис. 19).

Далее необходимо правой кнопкой по окну редактирования открыть контекстное меню и левой кнопкой выбрать мастер классов ClassWizrd (рис. 20).

В панели мастера классов MFC ClassWizard на закладке Member Vaiables выбрать IDC_EDIT1, нажать на кнопку Add Variable и в появившейся панели Add Member Variable добавить edit1 к имени переменной m_ и в результате получится имя переменной m_edit1. Выбрать тип переменной. Нажать на кнопку OK (рис. 21).

Ввод-вывод информации обеспечивается функцией UpdateData (argument) (рис. 22).

Функция UpdateData(argument) в зависимости от значения аргумента позволяет направить информацию из программы в окно редактирования (false) или направить информацию из окна редактирования в программу (true).

Рис. 19. Панель Controls

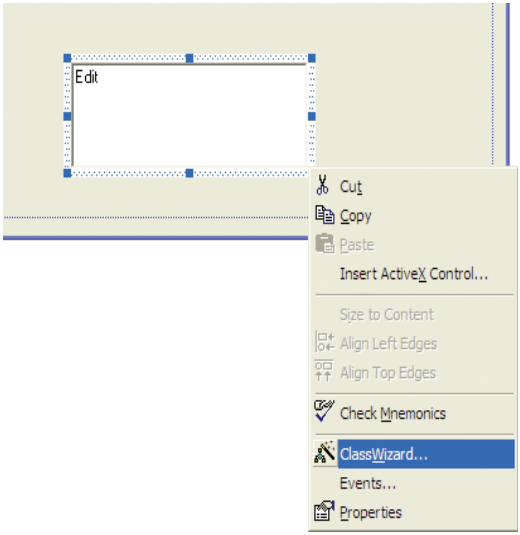


Рис. 20. Проектируемая диалоговая панель с размещенным на ней окном редактирования

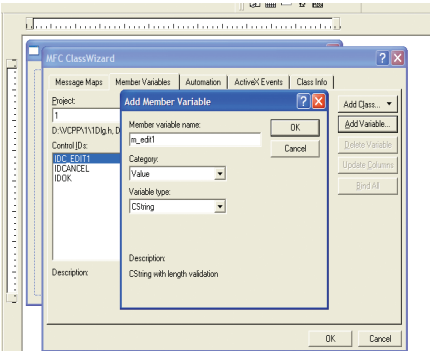


Рис. 21. Работа с мастером классов ClassWizard

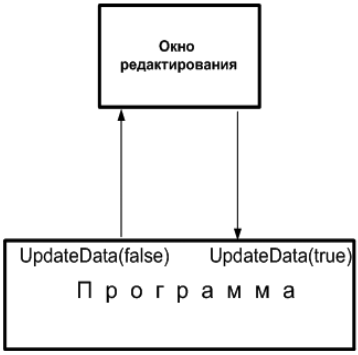


Рис. 22. Ввод и вывод информации

3.4. Добавление к проекту ActiveX элемента

Для размещения на проектируемой диалоговой панели таблицы выбираем Project, затем Add to project, а затем Components and Controls (рис. 23).

Из предложенных двух вариантов выбираем Registered ActiveX Controls (рис. 24) и нажимаем на кнопку Insert.

Выбрать из списка Microsoft FlexGrid Control, version 6.0(SP6) и нажать кнопку Insert (рис. 25).

Подтвердить вставку элемента кнопкой OK (рис. 26).

Подтвердить еще раз нажатием на кнопку OK (рис. 27).

В завершение на панели Components and Controls Galery нажать Close (рис. 28).

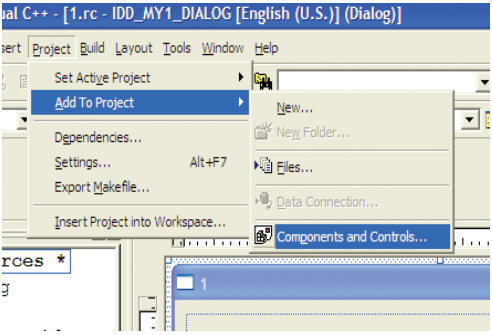


Рис. 23. Добавление в проект библиотеки ActiveX элементов

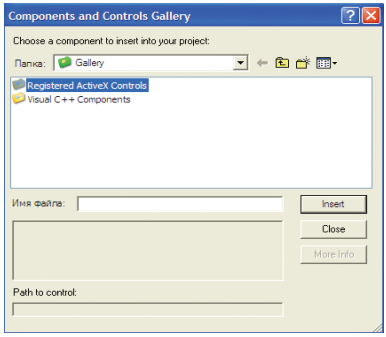


Рис. 24. Папки панели Components and Controls Galery

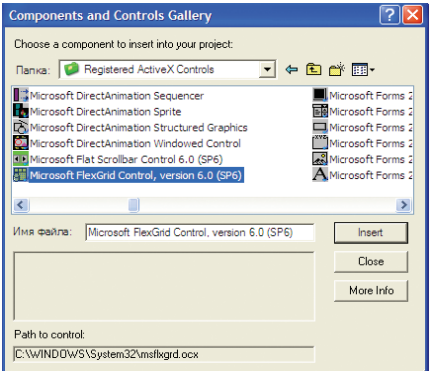


Рис. 25. Элементы папки Registered ActiveX Controls панели Components and Controls Galery

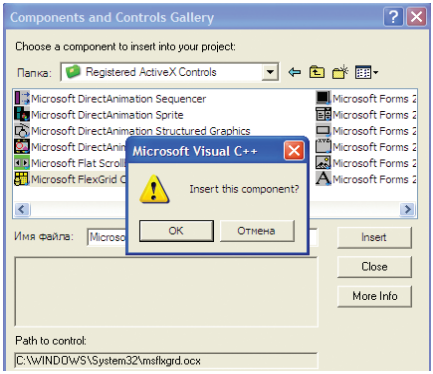


Рис. 26. Панель Components and Controls Galery

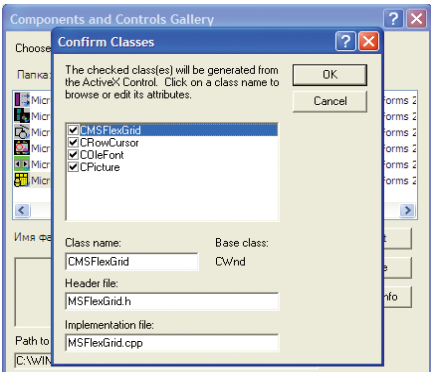


Рис. 27. Панель Confirm Classes

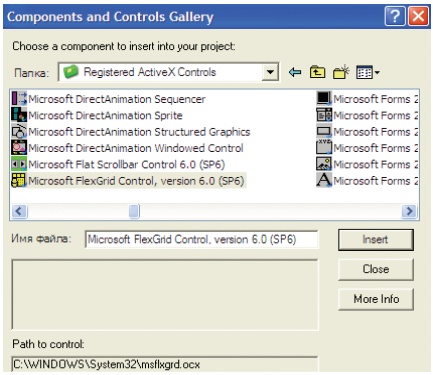


Рис. 28. Панель Components and Controls Galery

3.5. Создание таблицы FlexGrid

Для размещения на проектируемой диалоговой панели таблицы выбираем на панели Controls появившуюся пиктограмму таблицы (рис. 29).

И переносим таблицу на проектируемую диалоговую панель установив ее размеры (рис. 30).

Для таблицы зададим свойства. Для этого с помощью мастера классов ClassWizard на закладке Message Maps выбираем идентификатор таблицы IDC_MSFLEXGRID1 (рис. 31).

Выбираем Message типа Click и нажимаем на кнопку AddFunction и добавляем функцию OnClickMSFlexgrid1 (рис. 32).

Точно так же добавляем в таблицу свойство KeyPress и добавляем функцию OnKeyPressMSFlexgrid1 (рис. 33).

Чтобы можно было с созданной таблицей общаться с помощью программы присваиваем ей с помощью мастера классов на закладке Member Variables имя m_FG1. Для этого надо выделить таблицу и войти в мастер классов, нажать на кнопку Add Variable. В строке Member Variable Name добавить FG1. (рис. 34).

Данные в таблицу смогут поступать с помощью функций OnKeyPressMsFlexGrid(short Far*Key Ascii) и SetTextMatrix() (рис. 35).

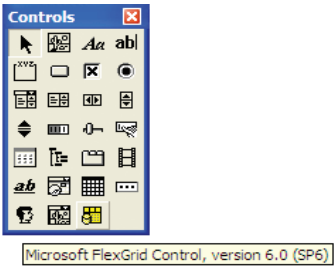


Рис. 29. Панель Controls и появившаяся пиктограмма таблицы

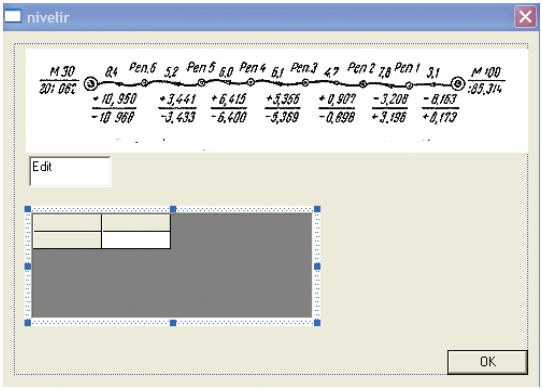


Рис. 30. Проектируемая диалоговая панель с таблицей типа GlexGrid

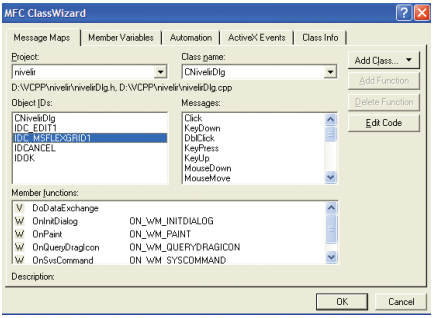


Рис. 31. Выбор таблицы в мастере классов ClassWizard

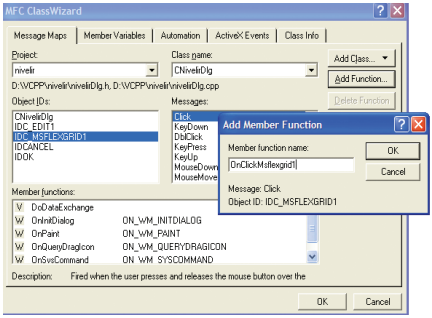


Рис. 32. Выбор свойства таблицы Click

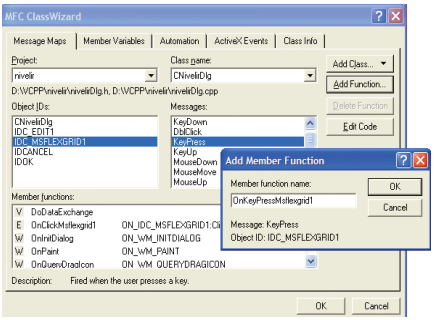


Рис. 33. Выбор свойства таблицы KeyPress

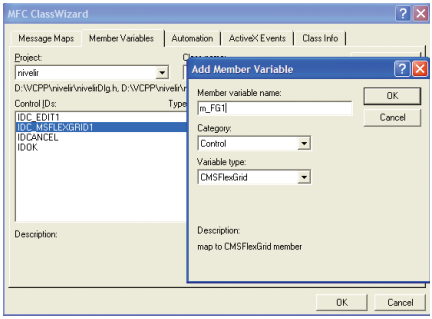


Рис. 34. Присваивание таблице своего уникального имени

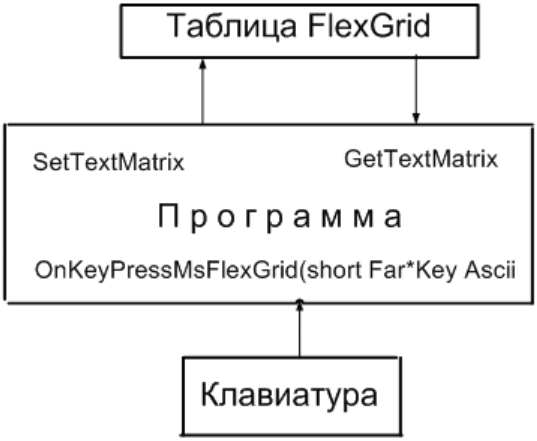


Рис. 35. Ввод-вывод информации в таблицу

3.6. Начальные установки программы

Установки начальные (таблицы должны быть невидимыми при запуске приложения).

Фрагмент программы «инициализация»

```
BOOL CNivelir_4Dlg::OnInitDialog()
{       CDialog::OnInitDialog();
        // Set the icon for this dialog. The framework does this automatically
        // when the application›s main window is not a dialog
        SetIcon(m_hIcon, TRUE);                         // Set big icon
        SetIcon(m_hIcon, FALSE);                // Set small icon
```

```
        // TODO: Add extra initialization here
        // делаем невидимыми таблицы, текст и кнопку
        GetDlgItem(IDC_MSFLEXGRID1)->ShowWindow(false);
        GetDlgItem(IDC_TEXT1)->ShowWindow(false);
        GetDlgItem(IDC_MSFLEXGRID2)->ShowWindow(false);
        GetDlgItem(IDC_BUTTON2)->ShowWindow(false);
        // установка параметров шрифта
        m_Font.CreateFont(25, 0, 0, 0, FW_NORMAL, FALSE,FALSE,0,0,0,0,0,0,
        «Tahoma»);
        GetDlgItem(IDC_TEXT)->SetFont(&m_Font);
        GetDlgItem(IDC_TEXT2)->SetFont(&m_Font);
    // установить текст
    SetDlgItemText(IDC_TEXT, «Вводим высоты двух твердых пунктов (марок)
и количество измерений»);

                return TRUE; // return TRUE unless you set the focus to a control
    }
```

3.7. Создание кнопки и обработчика события Click

С помощью мастера создаем кнопку (рис. 36) выбираем событие «одиночный клик» (рис. 37).

Фрагмент программы «события при нажатии на Button1» (за кнопкой 1 закрепляем программно действия)

```
void CNivelir_4Dlg::OnButton1()
{
        // TODO: Add your control notification handler code here
        //SetDlgItemText(IDC_ST_TEXT, «Ввести значения превышений»);

        char s[10]; int l;

        m_Font.CreateFont(25, 0, 0, 0, FW_NORMAL, FALSE,FALSE,0,0,0,0,0,0,
 «Tahoma»);
        GetDlgItem(IDC_TEXT)->ShowWindow(false);
        GetDlgItem(IDC_TEXT1)->ShowWindow(true);
        GetDlgItem(IDC_TEXT1)->SetFont(&m_Font);
        SetDlgItemText(IDC_TEXT1, «Вводим измеренные превышения»);

        //SetDlgItemText(IDC_TEXT, «Вводим измеренные значения
превышений прямые и обратные и длины участков»);
        GetDlgItem(IDC_MSFLEXGRID1)->ShowWindow(true);
        GetDlgItem(IDC_BUTTON2)->ShowWindow(true);
        UpdateData(true);
```

```
        n=m_edit3;
        //размеры таблицы
        m_FG1.SetCols(4);
        m_FG1.SetRows(n+1);
        //высота ячеек
        m_FG1.SetRowHeight(0,800);
        for(l=1; l<n+1; l++)
                m_FG1.SetRowHeight(l,350);
        //размер шрифта и толщина символов
        m_FG1.SetRow(0);
        for(l=0; l<4; l++)
        {
        m_FG1.SetCol(l);
        m_FG1.SetCellFontSize(9);
        m_FG1.SetCellFontBold(true);
        }
        //размеры шапки таблицы
        m_FG1.SetColWidth(0,400);
        m_FG1.SetColWidth(1,1300);
        m_FG1.SetColWidth(2,1300);
        m_FG1.SetColWidth(3,900);
        m_FG1.SetWordWrap(true);
        m_FG1.SetTextMatrix(0,0,»№\n п/п»);
        m_FG1.SetTextMatrix(0,1,»Превышение\nпрямое,\nм»);
        m_FG1.SetTextMatrix(0,2,»Превышение\nобратное,\nм»);
        m_FG1.SetTextMatrix(0,3,»Длина\nсекции,\nкм»);

        //нумерация строк
        j=0;
        for(i=0; i<n; i++)
        {
                itoa(i+1,s,10);
                m_FG1.SetTextMatrix(i+1,j,s);
        }

}
```

Объявление глобальных переменных

```
int i,j,n,A,B;
CString Stroka,Stroka1[20][20], Stroka2[20], Stroka3[20], Stroka4[20];
float data[20][20], Dohod[20], Sebest[20], Okup[20],Renta[20], dat[20];
```

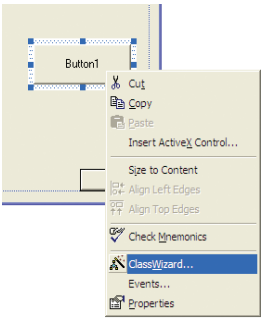


Рис. 36. Создаем кнопку (объект Button) и входим в мастер классов»

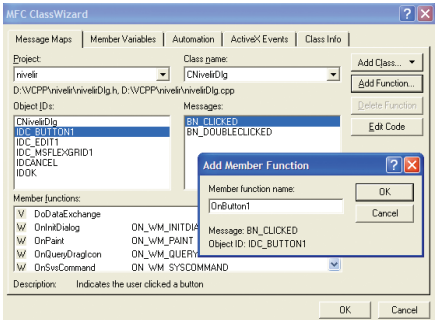


Рис. 37. Выбираем событие «одиночный клик»

Объявление оттенков цвета

```
const COLORREF CLOUDBLUE = RGB(128, 184, 223);
const COLORREF WHITE = RGB(255, 255, 255);
const COLORREF BLACK = RGB(1, 1, 1);
const COLORREF DKGRAY = RGB(128, 128, 128);
const COLORREF LTGRAY = RGB(192, 192, 192);
const COLORREF YELLOW = RGB(255, 255, 0);
const COLORREF DKYELLOW = RGB(128, 128, 0);
const COLORREF RED = RGB(255, 0, 0);
const COLORREF DKRED = RGB(128, 0, 0);
const COLORREF BLUE = RGB(0, 0, 255);
const COLORREF DKBLUE = RGB(0, 0, 128);
const COLORREF CYAN = RGB(0, 255, 255);
const COLORREF DKCYAN = RGB(0, 128, 128);
const COLORREF GREEN = RGB(0, 255, 0);
const COLORREF DKGREEN = RGB(0, 128, 0);
const COLORREF MAGENTA = RGB(255, 0, 255);
const COLORREF DKMAGENTA = RGB(128, 0, 128);
```

3.8. Создание второй кнопки и обработчика события Click

Фрагмент программы «события при нажатии на Button2» (за кнопкой 2 закрепляем программно действия)

```
void CNivelir_4Dlg::OnButton2()
{
    // TODO: Add your control notification handler code here
    float SummaPrev,PrevTeor,SummaDlin, MaxDelta; char a[20],b[20]=»»;
    int l,len,k1,k2,k5, d1,pd,d22;
```

```
CString z;
float d,d2, d2L, L, V, Vi, PredDi, fh;
char s[10];
GetDlgItem(IDC_TEXT1)->ShowWindow(false);

        // контроль заполненности таблицы исходных данных
        k2=0;
        for(i=1; i<n+1; i++)

        {
        k1=0;
        for(j=1; j<4; j++)

{
        Stroka1[i][j]=m_FG1.GetTextMatrix(i,j);
        len=strlen(Stroka1[i][j]);
        if(len==0) k1++;

}

        if(k1==4) k2++;
}
// чтение исходных данных из первой таблицы
for(i=1; i<n+1; i++)
for(j=1; j<4; j++)

{
        Stroka1[i][j]=m_FG1.GetTextMatrix(i,j);
        data[i][j]=atof(Stroka1[i][j]);
}

//таблица результатов

m_FG2.SetCols(8);
m_FG2.SetRows(n+1-k2);

m_FG2.SetRowHeight(0,350);

for(l=1; l<n+1-k2; l++)
m_FG2.SetRowHeight(l,350);

m_FG2.SetRow(0);

//размер шрифта и толщина символов
m_FG2.SetRow(0);
for(l=0; l<8; l++)
{
m_FG2.SetCol(l);
m_FG2.SetCellFontSize(9);
```

```
m_FG2.SetCellFontBold(true);
}

m_FG2.SetRowHeight(0,800);
m_FG2.ShowWindow(true);
m_FG2.SetColWidth(0,400);
m_FG2.SetColWidth(1,1250);
m_FG2.SetColWidth(2,1250);
m_FG2.SetColWidth(3,1250);
m_FG2.SetColWidth(4,1250);
m_FG2.SetColWidth(5,1250);
m_FG2.SetColWidth(6,1250);
m_FG2.SetColWidth(7,1250);
m_FG2.SetCellFontBold(true);
m_FG2.SetCellFontSize(9);

m_FG2.SetTextMatrix(0,0,»№\n п/п»);
m_FG2.SetTextMatrix(0,1,»Среднее превышение\nм»);
m_FG2.SetTextMatrix(0,2,»d,\nм»);
m_FG2.SetTextMatrix(0,3,»доп.превыш,\nм»);
m_FG2.SetTextMatrix(0,4,»d2,\nкв.м»);
m_FG2.SetTextMatrix(0,5,»d2/L,\nкв.м/км»);
m_FG2.SetTextMatrix(0,6,»Поправка V,\nм»);
m_FG2.SetTextMatrix(0,7,»Исправл.прев. V,\nм»);
m_FG2.SetWordWrap(true);

//нумерация строк таблицы результатов
j=0;
for(i=0; i<n-k2; i++)
{
    itoa(i+1,s,10);
    m_FG2.SetTextMatrix(i+1,j,s);
}
/*
CString W=»Сумма средних превышений практических=», W1=» Теоретическая разница высот=»,
        W2=» Допустимая невязка в мм=»,W3=» Предельная невязка=»,W4;
    //char buf[50];

    gcvt (SummaPrev,5,buf);
    W=W+buf;
    //MessageBox(W);
    PrevTeor=m_edit2-m_edit1;
    gcvt(PrevTeor,5,buf);
    W1=W1+buf;
    //MessageBox(W1);
```

```
        DopNevaz=(SummaPrev-PrevTeor);//*1000;
        gcvt(abs(DopNevaz),5,buf);
        W2=W2+buf;
        //MessageBox(W2);
        SummaDlin=10*pow(SummaDlin,.5);
        gcvt(SummaDlin,5,buf);
        W3=W3+buf;
        W4=W+W1+W2+W3;
        MessageBox(W4);
                        //if(DopNevaz<SummaDlin)// проверка корректности
исходных данных

                //MessageBox»Б у д е м у р а в н и в а т ь»);
        //else MessageBox(«Н а д о в с е п е р е м е р и т ь»);
    */
    SummaPrev=SummaDlin=MaxDelta=k5=0;
    for(i=1; i<n+1; i++)
    {
        sredn[i]=(fabs(data[i][1])+fabs(data[i][2]))/2;
        if(data[i][1]<0) sredn[i]=sredn[i]*(-1);
        SummaPrev+=sredn[i];
        SummaDlin+=data[i][3];
    }
        PrevTeor=m_edit2-m_edit1;
        fh=SummaPrev-PrevTeor;//значение невязки

        for(i=1; i<n+1; i++)
        {
                gcvt(sredn[i],5,a);
                m_FG2.SetCol(1);
                m_FG2.SetRow(i);
                m_FG2.SetCellFontBold(true);
                m_FG2.SetCellFontSize(9);
                m_FG2.SetTextMatrix(i,1,a);
        d=data[i][1]+data[i][2]; d1=d*1000;
                gcvt(d1,5,a);
                m_FG2.SetCol(2);
                m_FG2.SetRow(i);
                m_FG2.SetCellFontBold(true);
                m_FG2.SetCellFontSize(9);
                m_FG2.SetTextMatrix(i,2,a);
        PredDi=10*pow(data[i][3],.5); pd=PredDi;
                gcvt(pd,5,a);
                m_FG2.SetCol(3);
                m_FG2.SetRow(i);
```

```
                m_FG2.SetCellFontBold(true);
                m_FG2.SetCellFontSize(9);
                m_FG2.SetTextMatrix(i,3,a);
    d2=d*d; d22=d2*1000000;
                gcvt(d22,5,a);
                m_FG2.SetCol(4);
                m_FG2.SetRow(i);
                m_FG2.SetCellFontBold(true);
                m_FG2.SetCellFontSize(9);
                m_FG2.SetTextMatrix(i,4,a);
    d2L=d2/data[i][3];
                gcvt(d2L*1000000,5,a);
                m_FG2.SetCol(5);
                m_FG2.SetRow(i);
                m_FG2.SetCellFontBold(true);
                m_FG2.SetCellFontSize(9);
                m_FG2.SetTextMatrix(i,5,a);
                L=0; L=L+data[i][3];
// расчет поправки
        V=-fh*data[i][3]/SummaDlin;
        //V=-DopNevaz*data[i][3]/SummaDlin;
        gcvt(V*1000,5,a);
        m_FG2.SetCol(6);
        m_FG2.SetRow(i);
        m_FG2.SetCellFontBold(true);
        m_FG2.SetCellFontSize(9);
        m_FG2.SetTextMatrix(i,6,a);
// расчет исправленного превышения
        Vi=sredn[i]+V;
        gcvt(Vi,5,a);
        m_FG2.SetCol(7);
        m_FG2.SetRow(i);
        m_FG2.SetCellFontBold(true);
        m_FG2.SetCellFontSize(9);
        m_FG2.SetTextMatrix(i,7,a);
                if((d==PredDi)||(d>PredDi)) k5+=1;
}
if(k5>0) MessageBox(«Нельзя!!!»);
m_edit4=SummaDlin;
m_edit5=SummaPrev;
m_edit6=fh;
UpdateData(false);
}
```

Карта памяти

```
BEGIN_EVENTSINK_MAP(CNivelir_4Dlg, CDialog)
    //{{AFX_EVENTSINK_MAP(CNivelir_4Dlg)
        ON_EVENT(CNivelir_4Dlg, IDC_MSFLEXGRID1, -600 /* Click */,
OnClickMsflexgrid1, VTS_NONE)
        ON_EVENT(CNivelir_4Dlg, IDC_MSFLEXGRID1, -603 /* KeyPress */,
OnKeyPressMsflexgrid1, VTS_PI2)
        //}}AFX_EVENTSINK_MAP
END_EVENTSINK_MAP()
```

Создаем возможность окрашивания текста в определенный цвет

```
HBRUSH CNivelir_4Dlg::OnCtlColor(CDC* pDC, CWnd* pWnd, UINT nCtlColor)
{
    HBRUSH hbr = CDialog::OnCtlColor(pDC, pWnd, nCtlColor);
    // можно добавить: if ( nCtlColor == CTLCOLOR_STATIC )
        if ( pWnd->GetDlgCtrlID() == IDC_TEXT ) {
                pDC->SetTextColor(RGB(255, 0, 0));
                        pDC->SetBkMode(TRANSPARENT);}
        if ( pWnd->GetDlgCtrlID() == IDC_TEXT1 ) {
                pDC->SetTextColor(RGB(255, 0, 255));
                        pDC->SetBkMode(TRANSPARENT); }
        if ( pWnd->GetDlgCtrlID() == IDC_TEXT2 ) {
                pDC->SetTextColor(RGB(0, 125, 125));
                        pDC->SetBkMode(TRANSPARENT);}
    return hbr;
}
```

Обработка клика мышкой

```
void CNivelir_4Dlg::OnClickMsflexgrid1()
{
        //Обработка клика мышкой
        Stroka=» «;
        int h=m_FG1.GetCol();
        int b=m_FG1.GetRow();
        A=h; B=b;

        for(i=1; i<n+1; i++)
        for(j=1; j<4; j++)
        {
                m_FG1.SetCol(j);
                m_FG1.SetRow(i);
                m_FG1.SetCellBackColor(RGB(255,255,255));
        }
```

```
        m_FG1.SetCol(A);
        m_FG1.SetRow(B);
        m_FG1.SetCellBackColor(RGB(200,255,200));
        m_FG1.SetCellFontBold(true);
        m_FG1.SetCellFontSize(12);
    }
```

Запись из буфера клавиатуры в ячейку таблицы

```
void CNivelir_4Dlg::OnKeyPressMsflexgrid1(short FAR* KeyAscii)
{
        //Запись из буфера клавиатуры в ячейку таблицы
        Stroka+=FAR * KeyAscii;
        m_FG1.SetTextMatrix(B,A,Stroka);
}
```

Контрольные вопросы:

а) из каких компонент строится структура программы на C++ (подпрограммы-процедуры, подпрограммы-функции, функции)?

б) в чем разница между локальными и глобальными переменными?

в) назначение функции UpdateData();

г) назначение функции MessageBox;

д) назначение функции OnClickMsflexgrid1();

е) назначение функции OnKeyPressMsflexgrid1(short FAR* KeyAscii).

ж) назначение функции SetTextMatrix;

з) назначение функции GetTextMatrix.

3.9. Запуск и работа с приложением

После запуска приложения диалоговая панель имеет следующий вид (рис. 38).

Далее надо ввести в диалоговые окна высоты начального и конечного пунктов, количество промежуточных пунктов и нажать на кнопку с надписью «Вывести таблицу для ввода исходных данных». В результате диалоговая панель приобретет следующий вид (рис. 39).

В таблицу необходимо ввести превышения в прямом и обратном направлениях и длины участков хода. В результате получим следующий вид диалоговой панели (рис. 40).

После этого для оценки надежности и качества исходных данных надо нажать на кнопку «Оценить надежность хода». Если данные некорректны, то система выдаст сообщение «Необходимо произвести замеры повторно». Если с качеством измерений все в порядке, то получим диалоговую панель в виде (рис. 41).

Предлагается сопоставить результаты, полученные с помощью разработанного приложения (табл. 3), и результаты, полученные из практикума [1]:

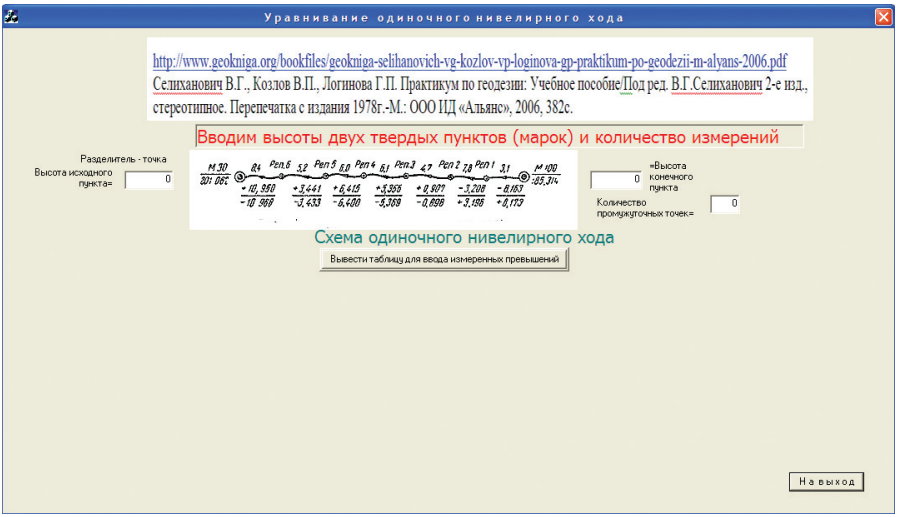


Рис. 38. Диалоговая панель после запуска

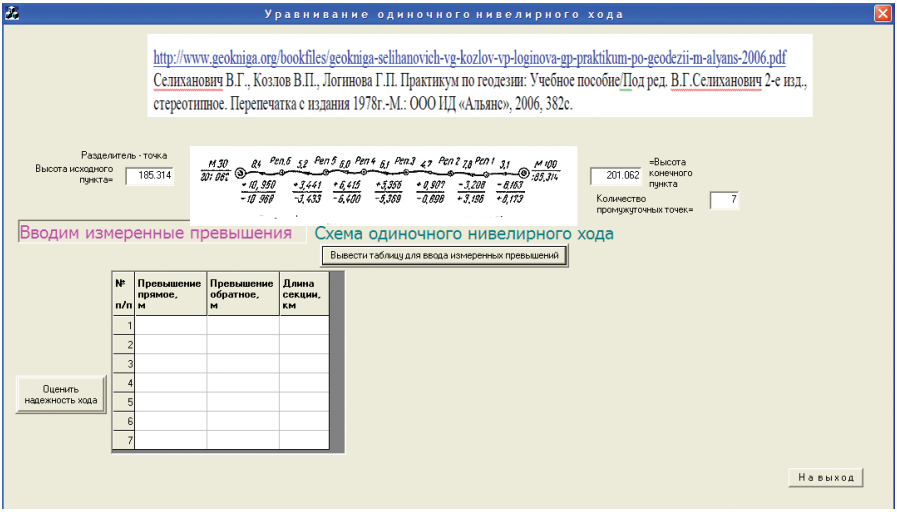
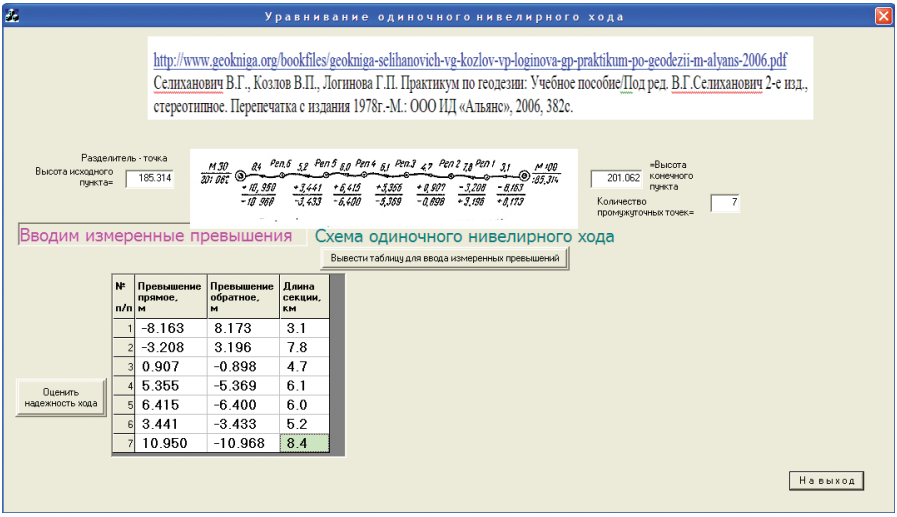
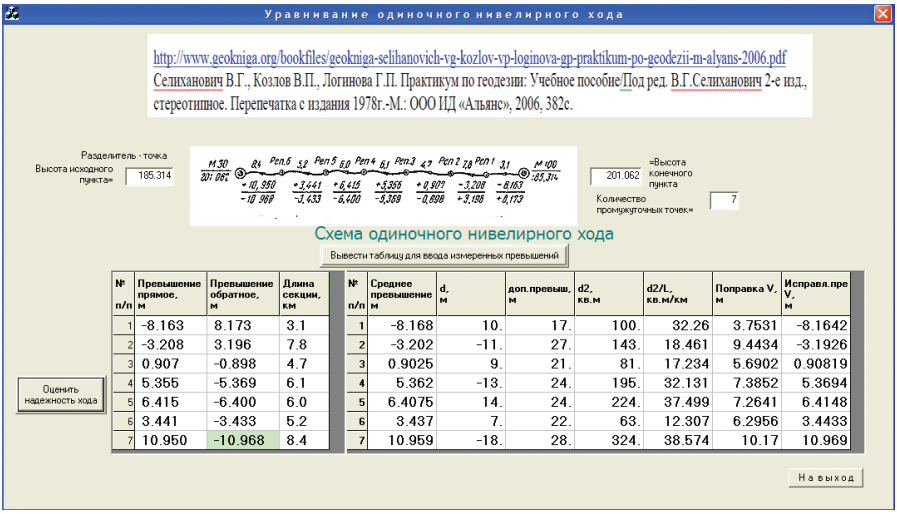


Рис. 39. Диалоговая панель с введенными начальными высотами и количеством промежуточных пунктов и выведенной таблицей для ввода исходных данных



№ п/п	Превышение прямое, м	Превышение обратное, м	Длина секции, км
1	-8.163	8.173	3.1
2	-3.208	3.196	7.8
3	0.907	-0.898	4.7
4	5.355	-5.369	6.1
5	6.415	-6.400	6.0
6	3.441	-3.433	5.2
7	10.950	-10.968	8.4

Рис. 40. Диалоговая панель с введенными исходными данными



№ п/п	Превышение прямое, м	Превышение обратное, м	Длина секции, км
1	-8.163	8.173	3.1
2	-3.208	3.196	7.8
3	0.907	-0.898	4.7
4	5.355	-5.369	6.1
5	6.415	-6.400	6.0
6	3.441	-3.433	5.2
7	10.950	-10.968	8.4

№ п/п	Среднее превышение м	d, м	доп.превыш, м	d2, кв.м	d2/L, кв.м/км	Поправка V, м	Исправл.пре V, м
1	-8.168	10.	17.	100.	32.26	3.7531	-8.1642
2	-3.202	-11.	27.	143.	18.461	9.4434	-3.1926
3	0.9025	9.	21.	81.	17.234	5.6902	0.90819
4	5.362	-13.	24.	195.	32.131	7.3852	5.3694
5	6.4075	14.	24.	224.	37.499	7.2641	6.4148
6	3.437	7.	22.	63.	12.307	6.2956	3.4433
7	10.959	-18.	28.	324.	38.574	10.17	10.969

Рис. 41. Диалоговая панель с результатами

Таблица 3

Результаты уравнивания одиночного нивелирного хода

№ п/п	Среднее превышение, м	*d*, м	Дополнительное превышение, м	*d*2, м^2	*d*2/*L*, м^2/км	Поправка *V*, м	Исправленное превышение *V*, м
1	–8,168	10,	17,	100,	32,26	3,7531	–8,1642
2	–3,202	–11,	27,	143,	18,461	9,4434	–3,1926
3	0,9025	9,	21,	81,	17,234	5,6902	0,90819
4	5,362	–13,	24,	195,	32,131	7,3852	5,3694
5	6,4075	14,	24,	224,	37,499	7,2641	6,4148
6	3,437	7,	22,	63,	12,307	6,2956	3,4433
7	10,959	–18,	28,	324,	38,574	10,17	10,969

4. ТЕСТИРОВАНИЕ РАЗРАБОТАННОЙ ПРОГРАММЫ

Выполнялось на основе примера в практикуме [1]. При проверке результаты работы программы совпали с рассчитанными с помощью калькулятора данными в таблице практикума. Поэтому диагностика и отладка программы не производились.

Контрольные вопросы:

а) какие Вы знаете методы и стратегии тестирования?
б) назначение диагностики программы;
в) какие Вы знаете методы отладки программы?

Глоссарий

Нивели́рование — вид геодезических работ, заключающийся в определении высот точек земной поверхности относительно уровенной поверхности, принятой за начальную.

Нивелирный ход — геодезический ход, выполненный с применением нивелира.

Уравнивание — процесс нахождения по результатам измерений согласованных оценок измерявшихся величин или их функций, выполняемый при математической обработке результатов геодезических измерений.

Репе́р — геодезический знак на местности с известной абсолютной высотой, определяемой из нивелирования.

Алгори́тм — набор инструкций, описывающих порядок действий для достижения некоторого результата.

Список литературы

1. *Селиханович В.Г., Козлов В.П., Логинова Г.П.* Практикум по геодезии: Учебное пособие / Под ред. В.Г. Селиханович, 2-е изд., стереотипное. — М.: ООО ИД «Альянс», 2006. — 382с. (http://www.geokniga.org/bookfiles/geokniga-selihanovich-vg-kozlov-vp-loginova-gp-praktikum-po-geodezii-m-alyans-2006.pdf)

2. *Березин Б.И., Березин С.Б.* Начальный курс С и С++. — М.: «Диалог» МИФИ, 1996. — 288 с.

3. *В.П. Скисов.* VC++6.0 и MFC: Справочное пособие. Вып. 1. Из-во «Мозырская укрупненная типография», 1999. — 190 с.

4. *А.В. Крячков, И.В. Сухонина, В.К. Томшин.* Программирование на С и С++: Практикум. — М.: Радио и связь, 1997. — 344 с.

5. *К. Джамса*. Учимся программировать на языке С++. — М.: Мир, 1999. — 320.

6. *К. Паппас, У. Мюррей.* Программирование на С и С++. К.г. Издательская группа BHV, 2000. — 320 с.

7. *Н. Гуревич, О. Гуревич.* Visual C++ 5. — М.: ЗАО «Издательство БИНОМ», 1998. — 624 с.

8. *Дж. Мюллер.* Visual C++ 5. — СПб: ВНУ–Санкт-Петербург, 1998. — 720 с.

9. *Грегори К.* Использование Visual C • 5: Специальное издание. — К.: Диалектика, 1997. — 816 с.

10. *И.И. Лонский.* Пакет программ математической обработки матриц: Свидетельство №2001610022. Роспатент, 12 января 2001 г.

11. *А.С. Фарафонов.* Программирование на языке высокого уровня [Электронный ресурс]: Методические указания к проведению лабораторных работ по курсу «Программирование» / А.С. Фарафонов. — Электрон. текстовые данные. — Липецк: Липецкий государственный технический университет, ЭБС АСВ, 2013. — 32 с.— 2227–8397. — Режим доступа: http://www.iprbookshop.ru/22912.html

12. Разработка Windows-приложений в среде программирования Visual Studio.Net [Электронный ресурс]: Учебно-методическое пособие по дисциплине «Информатика и программирование» /. — Электрон. текстовые данные. — М. : Московский технический университет связи и информатики, 2016. — 20 с. — 2227–8397. — Режим доступа: http://www.iprbookshop.ru/61536.html

13. *Н.И. Костюкова.* Программирование на языке Си [Электронный ресурс]: Методические рекомендации и задачи по программированию / Н.И. Костюкова. — Электрон. текстовые данные. — Новосибирск: Сибирское университетское издательство, 2017. — 160 с. — 978–5–379–02016–3. — Режим доступа: http://www.iprbookshop.ru/65289.html

14. Визуальное программирование на основе библиотеки MFC [Электронный ресурс]: Методические указания к лабораторным работам по курсу «Визуальное программирование» для студентов направления 09.03.02 Информационные системы и технологии /. — Электрон. текстовые данные. — Саратов: Вузовское образование, 2016. — 57 с. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/28324.html