Программный код модулей подсистемы

Листинг 1 – Программный код модуля формы элемента справочника «Абитуриенты»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// определяем доступность функции просмотра фотографии, если она добавлена
	Фотография = Объект.Фотография;
	Элемент = Элементы.Найти("ОткрытьФотографию");
	Если ЗначениеЗаполнено(Фотография) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// проверяем, есть ли среди добавленных ОП те, по которым требуется медицинская
справка
	// если есть, показываем это поле
	Для каждого СтрокаОП из Объект.ОбразовательныеПрограммы Цикл
		НеобходимостьМедицинскойСправки =
ПроверитьНаНеобходимостьМедицинскойСправки(СтрокаОП.ОбразовательнаяПрограмма);
		Если НеобходимостьМедицинскойСправки = Истина Тогда
			Элементы.РаботаСоСканКопиейМедицинскойСправки.Видимость = Истина;
			Прервать;
		КонецЕсли;
	КонецЦикла;

	// определяем доступность функции просмотра файла - медицинской справки, если он
добавлен
	МедицинскаяСправка = Объект.МедицинскаяСправка;
	Элемент = Элементы.Найти("ОткрытьФайл");
	Если ЗначениеЗаполнено(МедицинскаяСправка) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// при открытии формы уже созданного элемента проверяем,
	// надо ли показывать те или иные скрытые поля
	Если Не Объект.Гражданство =
ПредопределенноеЗначение("Справочник.Гражданства.РоссийскаяФедерация") Тогда
		Элементы.СНИЛС.Видимость = Ложь;

	Элементы.ПодтверждающийДокументИностранногоГражданинаИлиЛицаБезГражданства.Видимо
сть = Истина;
	КонецЕсли;
	Если Объект.ПреимущественноеПраво = Истина Тогда
		Элементы.ДокументПодтверждающийПреимущественноеПраво.Видимость = Истина;
	КонецЕсли;
	Если Объект.Квота = ПредопределенноеЗначение("Перечисление.Квоты.Особая") Тогда
		Элементы.ДокументПодтверждающийОсобоеПраво.Видимость = Истина;
	КонецЕсли;
	Если Объект.Квота = ПредопределенноеЗначение("Перечисление.Квоты.Целевая") Тогда
		Элементы.ДоговорОЦелевомОбучении.Видимость = Истина;
	КонецЕсли;
	Если Объект.Квота = ПредопределенноеЗначение("Перечисление.Квоты.Олимпиады")
Тогда
		Элементы.ДокументПодтверждающийСтатусПобедителяИлиПризёра.Видимость =
Истина;
	КонецЕсли;
КонецПроцедуры
```

Продолжение листинга 1

```
&НаСервереБезКонтекста
// функция проверки необходимости медицинской справки для ОП
Функция ПроверитьНаНеобходимостьМедицинскойСправки(ОбразовательнаяПрограмма)
	ОбразовательнаяПрограммаБД =
Справочники.ОбразовательныеПрограммы.НайтиПоКоду(ОбразовательнаяПрограмма.Код);
	Если ОбразовательнаяПрограммаБД.НеобходимостьМедицинскойСправки = Истина Тогда
		Возврат Истина;
	Иначе
		Возврат Ложь;
	КонецЕсли;
КонецФункции

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаРожденияПриИзменении(Элемент)
	Если Объект.Датарождения > ТекущаяДата() Тогда
		Предупреждение("Нельзя указать ещё не наступившую дату!");
		Объект.ДатаРождения = Неопределено;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
// настраиваем видимость документов, необходимых и допустимых в зависимости от
гражданства
Процедура ГражданствоПриИзменении(Элемент)
	Если Объект.Гражданство =
ПредопределенноеЗначение("Справочник.Гражданства.РоссийскаяФедерация") Тогда

	Элементы.ПодтверждающийДокументИностранногоГражданинаИлиЛицаБезГражданства.Видимо
сть = Ложь;
		Элементы.СНИЛС.Видимость = Истина;
	ИначеЕсли Объект.Гражданство =
ПредопределенноеЗначение("Справочник.Гражданства.ЛицоБезГражданства") Тогда

	Элементы.ПодтверждающийДокументИностранногоГражданинаИлиЛицаБезГражданства.Заголо
вок = "Подтверждающий документ лица без гражданства*";

	Элементы.ПодтверждающийДокументИностранногоГражданинаИлиЛицаБезГражданства.Видимо
сть = Истина;
		Элементы.СНИЛС.Видимость = Ложь;
	Иначе

	Элементы.ПодтверждающийДокументИностранногоГражданинаИлиЛицаБезГражданства.Заголо
вок = "Подтверждающий документ иностранного гражданина*";

	Элементы.ПодтверждающийДокументИностранногоГражданинаИлиЛицаБезГражданства.Видимо
сть = Истина;
		Элементы.СНИЛС.Видимость = Ложь;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
// если меняем уровень образования, на который поступает абитуриент, очищаем табличную
часть с ОП
Процедура УровеньОбразованияДляПоступленияПриИзменении(Элемент)
	Объект.ОбразовательныеПрограммы.Очистить();
КонецПроцедуры
```

Продолжение листинга 1

```
&НаКлиенте
// в зависимости от наличия преимущественного права, настраиваем видимость
подтверждающего документа
Процедура ПреимущественноеПравоПриИзменении(Элемент)
	Если Объект.ПреимущественноеПраво = Истина Тогда
		Элементы.ДокументПодтверждающийПреимущественноеПраво.Видимость = Истина;
	Иначе
		Элементы.ДокументПодтверждающийПреимущественноеПраво.Видимость = Ложь;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
// в зависимости от типа квоты настраиваем видимость подтверждающих документов
Процедура КвотаПриИзменении(Элемент)
	Если Объект.Квота = ПредопределенноеЗначение("Перечисление.Квоты.Особая") Тогда
		Элементы.ДокументПодтверждающийОсобоеПраво.Видимость = Истина;
		Элементы.ДоговорОЦелевомОбучении.Видимость = Ложь;
		Элементы.ДокументПодтверждающийСтатусПобедителяИлиПризёра.Видимость =
Ложь;
	ИначеЕсли Объект.Квота = ПредопределенноеЗначение("Перечисление.Квоты.Целевая")
Тогда
		Элементы.ДокументПодтверждающийОсобоеПраво.Видимость = Ложь;
		Элементы.ДоговорОЦелевомОбучении.Видимость = Истина;
		Элементы.ДокументПодтверждающийСтатусПобедителяИлиПризёра.Видимость =
Ложь;
	ИначеЕсли Объект.Квота = ПредопределенноеЗначение("Перечисление.Квоты.Олимпиады")
Тогда
		Элементы.ДокументПодтверждающийОсобоеПраво.Видимость = Ложь;
		Элементы.ДоговорОЦелевомОбучении.Видимость = Ложь;
		Элементы.ДокументПодтверждающийСтатусПобедителяИлиПризёра.Видимость =
Истина;
	Иначе
		Элементы.ДокументПодтверждающийОсобоеПраво.Видимость = Ложь;
		Элементы.ДоговорОЦелевомОбучении.Видимость = Ложь;
		Элементы.ДокументПодтверждающийСтатусПобедителяИлиПризёра.Видимость =
Ложь;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВступительныеИспытанияПриНачалеРедактирования(Элемент, НоваяСтрока,
Копирование)
	// записываем в поле "Год сдачи" по умолчанию текущий год
	Если НоваяСтрока И Не Копирование Тогда
		Элементы.ВступительныеИспытания.ТекущиеДанные.ГодСдачи = ТекущаяДата(); //
пишем именно дату, т. к. в поле "Год сдачи" хранится дата в виде "только год"
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
// проверка того, что количество добавленных образовательных программ не превышает
допустимое;
// допустимое значение зависит от того, на какой уровень образования поступает
абитуриент;
// эти допустимые значения хранятся в соответствующих константах
Процедура ОбразовательныеПрограммыПередНачаломДобавления(Элемент, Отказ, Копирование,
Родитель, Группа, Параметр)
	Если Объект.УровеньОбразованияДляПоступления =
ПредопределенноеЗначение("Перечисление.УровниОбразования.БакалавриатСпециалитет") Тогда
```

Продолжение листинга 1

```
            Если Объект.ОбразовательныеПрограммы.Количество() >=
ПолучитьКонстантуМаксимальноеКоличествоНаправленийБакалавриатСпециалитет() Тогда
                  Сообщить("Образовательных программ бакалавриата/специалитета для
участия в конкурсе не может быть больше " +
ПолучитьКонстантуМаксимальноеКоличествоНаправленийБакалавриатСпециалитет() + "!");
                  Отказ = Истина;
            КонецЕсли;
      ИначеЕсли Объект.УровеньОбразованияДляПоступления =
ПредопределенноеЗначение("Перечисление.УровниОбразования.Магистратура") Тогда
            Если Объект.ОбразовательныеПрограммы.Количество() >=
ПолучитьКонстантуМаксимальноеКоличествоНаправленийМагистратура() Тогда
                  Сообщить("Образовательных программ магистратуры для участия в
конкурсе не может быть больше " +
ПолучитьКонстантуМаксимальноеКоличествоНаправленийМагистратура() + "!");
                  Отказ = Истина;
            КонецЕсли;
      КонецЕсли;
КонецПроцедуры

&НаСервереБезКонтекста
Функция ПолучитьКонстантуМаксимальноеКоличествоНаправленийБакалавриатСпециалитет()
      Возврат
Константы.МаксимальноеКоличествоНаправленийБакалавриатаСпециалитета.Получить();
КонецФункции

&НаСервереБезКонтекста
Функция ПолучитьКонстантуМаксимальноеКоличествоНаправленийМагистратура()
      Возврат Константы.МаксимальноеКоличествоНаправленийМагистратуры.Получить();
КонецФункции

&НаСервереБезКонтекста
// Функция нужна, чтобы показать поле медицинской справки, если добавлена ОП, где эта
справка требуется
Функция ОбразовательныеПрограммыПриИзмененииНаСервере(ОбразовательныеПрограммы)
      Для каждого СтрокаОП из ОбразовательныеПрограммы Цикл
            ОбразовательнаяПрограмма =
Справочники.ОбразовательныеПрограммы.НайтиПоКоду(СтрокаОП.ОбразовательнаяПрограмма);
            Если ОбразовательнаяПрограмма.НеобходимостьМедицинскойСправки = Истина
Тогда
                  Возврат Истина;
            КонецЕсли;
      КонецЦикла;
      Возврат Ложь;
КонецФункции

&НаКлиенте
// получение результата выполнения описанной выше функции
Процедура ОбразовательныеПрограммыПриИзменении(Элемент)
      ОбразовательныеПрограммы = Объект.ОбразовательныеПрограммы;
      НеобходимостьСправки =
ОбразовательныеПрограммыПриИзмененииНаСервере(ОбразовательныеПрограммы);
      Если НеобходимостьСправки = Истина Тогда
            Элементы.РаботаСоСканКопиейМедицинскойСправки.Видимость = Истина;
      Иначе
            Элементы.РаботаСоСканКопиейМедицинскойСправки.Видимость = Ложь;
      КонецЕсли;
КонецПроцедуры

&НаСервере
// вызов процедуры проверки заполнения из модуля объекта
```

Продолжение листинга 1

```
Процедура ПередЗаписьюНаСервере(Отказ, ТекущийОбъект, ПараметрыЗаписи)
	Документ = РеквизитФормыВЗначение("Объект");
	Документ.ПроверитьЗаполнение();
КонецПроцедуры

&НаСервереБезКонтекста
// добавление записи с баллами абитуриента в регистр сведений после создания записи об
абитуриенте
Процедура ПослеЗаписиНаСервере(КодАбитуриента, ИндивидуальныеДостиженияАбитуриента,
ОбразовательныеПрограммыАбитуриента, ВступительныеИспытанияАбитуриента)
	СуммаБалловЗаИД = 0;
	// пройдёмся по каждому ИД абитуриента, найдём через ИБ соответствующий ему балл
и сложим баллы за ИД
	Для каждого СтрокаИД из ИндивидуальныеДостиженияАбитуриента Цикл
		Запрос = Новый Запрос;
		Запрос.Текст =
			"ВЫБРАТЬ
			|	ИндивидуальныеДостижения.Баллы КАК Баллы
			|ИЗ
			|	Справочник.ИндивидуальныеДостижения КАК
ИндивидуальныеДостижения
			|ГДЕ
			|	ИндивидуальныеДостижения.Наименование = &Наименование";
		Запрос.УстановитьПараметр("Наименование", Строка(СтрокаИД.Наименование));
		РезультатЗапроса = Запрос.Выполнить();
		ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
		ВыборкаДетальныеЗаписи.Следующий();
		СуммаБалловСтроки = ВыборкаДетальныеЗаписи.Баллы;
		СуммаБалловЗаИД = СуммаБалловЗаИД + СуммаБалловСтроки;
	КонецЦикла;

	// пройдёмся по каждой образовательной программе абитуриента
	Для каждого СтрокаОП из ОбразовательныеПрограммыАбитуриента Цикл
		// создаём новую запись в регистре сведений для текущей в цикле ОП
абитуриента
		НоваяЗапись = РегистрыСведений.БаллыАбитуриентов.СоздатьМенеджерЗаписи();
		НоваяЗапись.Абитуриент =
Справочники.Абитуриенты.НайтиПоКоду(КодАбитуриента);
		НоваяЗапись.ОбразовательнаяПрограмма =
Справочники.ОбразовательныеПрограммы.НайтиПоКоду(СтрокаОП.ОбразовательнаяПрограмма);
		НоваяЗапись.СуммаБалловЗаИД = СуммаБалловЗаИД;

		СуммаБалловДляРегистраСведений = СуммаБалловЗаИД;
		ТекущаяОбразовательнаяПрограмма = СтрокаОП.ОбразовательнаяПрограмма;

		// определим вступительные испытания данной образовательной программы
		Запрос = Новый Запрос;
		Запрос.Текст =
			"ВЫБРАТЬ
			|	ОбразовательныеПрограммы.ВступительныеИспытания.(
			|		ВступительноеИспытание КАК ВступительноеИспытание,
			|		Приоритетность КАК Приоритетность
			|	) КАК ВступительныеИспытания
			|ИЗ
			|	Справочник.ОбразовательныеПрограммы КАК
ОбразовательныеПрограммы
			|ГДЕ
			|	ОбразовательныеПрограммы.Код = &Код";
		Запрос.УстановитьПараметр("Код", Строка(ТекущаяОбразовательнаяПрограмма));
		РезультатЗапроса = Запрос.Выполнить();
```

Продолжение листинга 1

```
            // пройдёмся по вступительным испытаниям из полученной выборки, чтобы
определить максимальную приоритетность ВИ текущей в цикле ОП
            МаксимальнаяПриоритетностьВИ = 0;
            ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
            Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
                  ВступительныеИспытания =
ВыборкаДетальныеЗаписи.ВступительныеИспытания.Выбрать();
                  Пока ВступительныеИспытания.Следующий() Цикл
                        Если ВступительныеИспытания.Приоритетность >
МаксимальнаяПриоритетностьВИ Тогда
                              МаксимальнаяПриоритетностьВИ =
ВступительныеИспытания.Приоритетность;
                        КонецЕсли;
                  КонецЦикла;
            КонецЦикла;

            // создадим массив, в котором будут храниться баллы абитуриента за ВИ с
разделением по приоритетностям ВИ в рамках данной ОП
            БаллыЗаВИПоПриоритетностям = Новый Массив(8);

            // пройдёмся по вступительным испытаниям из полученной выборки
            ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
            Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
                  // будем поочерёдно обрабатывать ВИ каждой приоритетности от 1 до
максимальной приоритетности ВИ
                  Для н = 0 по МаксимальнаяПриоритетностьВИ - 1 Цикл
                        НаибольшийБаллЗаВИТекущейПриоритетности = 0;

                        ВступительныеИспытания =
ВыборкаДетальныеЗаписи.ВступительныеИспытания.Выбрать();
                        Пока ВступительныеИспытания.Следующий() Цикл
                              Если ВступительныеИспытания.Приоритетность = н + 1
Тогда // берём только ВИ текущей в цикле приоритетности
                                    // найдём это ВИ у абитуриента
                                    Для каждого СтрокаВИ из
ВступительныеИспытанияАбитуриента Цикл
                                          Если СтрокаВИ.ВступительноеИспытание =
ВступительныеИспытания.ВступительноеИспытание И Число(СтрокаВИ.Баллы) >
НаибольшийБаллЗаВИТекущейПриоритетности Тогда
                                                // баллы записываем только в случае,
если они выше, чем баллы за другое ВИ такой же приоритетности

      НаибольшийБаллЗаВИТекущейПриоритетности = Число(СтрокаВИ.Баллы);
                                                БаллыЗаВИПоПриоритетностям[н] =
Число(СтрокаВИ.Баллы);
                                          КонецЕсли;
                                    КонецЦикла;
                              КонецЕсли;
                        КонецЦикла;

                        СуммаБалловДляРегистраСведений =
СуммаБалловДляРегистраСведений + НаибольшийБаллЗаВИТекущейПриоритетности;
                  КонецЦикла;
            КонецЦикла;

            НоваяЗапись.Баллы1Приоритета = БаллыЗаВИПоПриоритетностям[0];
            НоваяЗапись.Баллы2Приоритета = БаллыЗаВИПоПриоритетностям[1];
            НоваяЗапись.Баллы3Приоритета = БаллыЗаВИПоПриоритетностям[2];
            НоваяЗапись.Баллы4Приоритета = БаллыЗаВИПоПриоритетностям[3];
            НоваяЗапись.Баллы5Приоритета = БаллыЗаВИПоПриоритетностям[4];
```

Продолжение листинга 1

```
            НоваяЗапись.Баллы6Приоритета = БаллыЗаВИПоПриоритетностям[5];
            НоваяЗапись.Баллы7Приоритета = БаллыЗаВИПоПриоритетностям[6];
            НоваяЗапись.Баллы8Приоритета = БаллыЗаВИПоПриоритетностям[7];
            НоваяЗапись.СуммаБаллов = СуммаБалловДляРегистраСведений;
            НоваяЗапись.Записать();
      КонецЦикла;
КонецПроцедуры

&НаКлиенте
// передача параметров с клиента на сервер для описанной выше процедуры
Процедура ПослеЗаписи(ПараметрыЗаписи)
      КодАбитуриента = Объект.Код;
      ИндивидуальныеДостижения = Объект.ИндивидуальныеДостижения;
      ОбразовательныеПрограммы = Объект.ОбразовательныеПрограммы;
      ВступительныеИспытания = Объект.ВступительныеИспытания;
      ПослеЗаписиНаСервере(КодАбитуриента, ИндивидуальныеДостижения,
ОбразовательныеПрограммы, ВступительныеИспытания);
КонецПроцедуры

&НаКлиенте
// для реализации автозаполнения полей в дочерних формах передаём значения этих полей в
текущей (родительской) форме
Функция ПередачаЗначенийВФормыЭлемента() Экспорт
      ЗначенияПолейФормы = Новый Структура;
      ЗначенияПолейФормы.Вставить("ФИО", Объект.Наименование);
      ЗначенияПолейФормы.Вставить("Гражданство", Объект.Гражданство);
      ЗначенияПолейФормы.Вставить("Пол", Объект.Пол);
      ЗначенияПолейФормы.Вставить("ДатаРождения", Объект.ДатаРождения);
      Если Не Элементы.ИндивидуальныеДостижения.ТекущиеДанные = Неопределено Тогда
            ЗначенияПолейФормы.Вставить("ИндивидуальноеДостижение",
Элементы.ИндивидуальныеДостижения.ТекущиеДанные.Наименование);
      КонецЕсли;
      Возврат ЗначенияПолейФормы;
КонецФункции

&НаКлиенте
Процедура ВыбратьФотографию(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл -
фотография абитуриента
      СтандартнаяОбработка = Ложь;
      Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
      Диалог.Заголовок = "Выберите файл";
      Диалог.Фильтр = "JPG, JPEG, PNG, PDF (*.jpg, *.jpeg, *.png,
*.pdf)|*.jpg;*.jpeg;*.png;*.pdf|" +
                              "Формат JPG (*.jpg)|*.jpg|" +
                              "Формат JPEG (*.jpeg)|*.jpeg|" +
                              "Формат PNG (*.png)|*.png|" + "Документ PDF
(*.pdf)|*.pdf";
      Диалог.МножественныйВыбор = Ложь;

      // если файл уже добавлен, получаем его путь и открываем Проводник по нему
      Если Не Объект.Фотография = Неопределено Тогда
            ФайлФотографии = Новый Файл(Объект.Фотография);
            Если ФайлФотографии.Существует() Тогда
                  Диалог.ПолноеИмяФайла = ФайлФотографии.ПолноеИмя;
                  Диалог.Каталог = ФайлФотографии.Путь;
            КонецЕсли;
      Иначе // открываем Проводник в корне диска С
            Диалог.ПолноеИмяФайла = "";
            Диалог.Каталог = "C:\";
      КонецЕсли;
```

Продолжение листинга 1

```
	Если Диалог.Выбрать() Тогда // если файл был выбран
		Объект.Фотография = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
		Элемент = Элементы.Найти("ОткрытьФотографию");
		Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФотографию(Команда) // файл - фотография абитуриента
	ФайлФотографии = Новый Файл(Объект.Фотография);
	Если ФайлФотографии.Существует() Тогда
		ЗапуститьПриложение(ФайлФотографии.ПолноеИмя);
	Иначе
		Сообщение = Новый СообщениеПользователю;
		Сообщение.Текст = "Файл не существует!";
		Сообщение.Поле = "Фотография";
		Сообщение.ПутьКДанным = "Объект";
		Сообщение.Сообщить();
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл -
медицинская справка
	СтандартнаяОбработка = Ложь;
	Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
	Диалог.Заголовок = "Выберите файл";
	Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
	Диалог.МножественныйВыбор = Ложь;

	// если файл уже добавлен, получаем его путь и открываем Проводник по нему
	Если Не Объект.МедицинскаяСправка = Неопределено Тогда
		ФайлPDF = Новый Файл(Объект.МедицинскаяСправка);
		Если ФайлPDF.Существует() Тогда
			Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
			Диалог.Каталог = ФайлPDF.Путь;
		КонецЕсли;
	Иначе // открываем Проводник в корне диска C
		Диалог.ПолноеИмяФайла = "";
		Диалог.Каталог = "C:\";
	КонецЕсли;

	// если файл был выбран
	Если Диалог.Выбрать() Тогда
		Объект.МедицинскаяСправка = Диалог.ПолноеИмяФайла; // записываем в
реквизит полный путь к файлу
		Элемент = Элементы.Найти("ОткрытьФайл");
		Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - медицинская справка
	ФайлPDF = Новый Файл(Объект.МедицинскаяСправка);
	Если ФайлPDF.Существует() Тогда
		ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
	Иначе
```

Окончание листинга 1

```
            Сообщение = Новый СообщениеПользователю;
            Сообщение.Текст = "Файл не существует!";
            Сообщение.Поле = "МедицинскаяСправка";
            Сообщение.ПутьКДанным = "Объект";
            Сообщение.Сообщить();
      КонецЕсли;
КонецПроцедуры
```

Листинг 2 – Программный код модуля объекта справочника «Абитуриенты»

```
Процедура ОбработкаПроверкиЗаполнения(Отказ, ПроверяемыеРеквизиты) Экспорт
      // Реквизиты формы элемента
      // в зависимости от гражданства нужно устанавливать обязательность заполнения
документов
      Если Гражданство <> Справочники.Гражданства.РоссийскаяФедерация Тогда

      ПроверяемыеРеквизиты.Добавить("ПодтверждающийДокументИностранногоГражданинаИлиЛиц
аБезГражданства");
      КонецЕсли;

      // аналогично по преимущественному праву и квотам
      Если ПреимущественноеПраво = Истина Тогда

      ПроверяемыеРеквизиты.Добавить("ДокументПодтверждающийПреимущественноеПраво");
      КонецЕсли;

      Если Квота = Перечисления.Квоты.Особая Тогда
            ПроверяемыеРеквизиты.Добавить("ДокументПодтверждающийОсобоеПраво");
      ИначеЕсли Квота = Перечисления.Квоты.Целевая Тогда
            ПроверяемыеРеквизиты.Добавить("ДоговорОЦелевомОбучении");
      ИначеЕсли Квота = Перечисления.Квоты.Олимпиады Тогда

      ПроверяемыеРеквизиты.Добавить("ДокументПодтверждающийСтатусПобедителяИлиПризёра")
;
      КонецЕсли;

      // проверка наличия медицинской справки, если среди образовательных программ
абитуриента есть такая, по которой необходима медицинская справка
      НеобходимостьСправки = Ложь;
      Для каждого СтрокаОП из ОбразовательныеПрограммы Цикл
            // найдём ОП в ИБ, чтобы понять, требуется ли мед. справка
            ОбразовательнаяПрограмма =
Справочники.ОбразовательныеПрограммы.НайтиПоКоду(СтрокаОП.ОбразовательнаяПрограмма);
            Если ОбразовательнаяПрограмма.НеобходимостьМедицинскойСправки = Истина
Тогда
                  НеобходимостьСправки = Истина;
                  Прервать;
            КонецЕсли;
      КонецЦикла;

      Если НеобходимостьСправки = Истина Тогда
            ПроверяемыеРеквизиты.Добавить("МедицинскаяСправка");
      КонецЕсли;

      // ВИ
      // проверка того, что добавленные вступительные испытания не повторяются
      Если ВступительныеИспытания.Количество() >= 2 Тогда // в проверке есть смысл,
если добавлено 2 и более ВИ
            // берём первую строку и сравниваем со всеми последующими, начиная со
второй
```

Продолжение листинга 2

```
            // ...
            // берём m-ю строку и сравниваем с последующими, начиная с (m+1)-й
            Для н = 0 по ВступительныеИспытания.Количество() - 2 Цикл
                  КоличествоПовторяющихсяСтрок = 0;

                  Для к = н + 1 по ВступительныеИспытания.Количество() - 1 Цикл
                        Если
ВступительныеИспытания.Получить(н).ВступительноеИспытание =
ВступительныеИспытания.Получить(к).ВступительноеИспытание Тогда
                              КоличествоПовторяющихсяСтрок =
КоличествоПовторяющихсяСтрок + 1;
                        КонецЕсли;
                  КонецЦикла;

                  // если после проверки строки нашлось совпадение - это уже повод
выдать ошибку
                  Если КоличествоПовторяющихсяСтрок > 0 Тогда
                        Сообщить("Добавленные вступительные испытания не должны
повторяться!");
                        Отказ = Истина;
                        Прервать;
                  КонецЕсли;
            КонецЦикла;
      КонецЕсли;

      // Проверка корректности введённых баллов за вступительные испытания (выше
минимального и меньше 100)
      Для каждого СтрокаВИ из ВступительныеИспытания Цикл
            Если ЗначениеЗаполнено(СтрокаВИ.ВступительноеИспытание) Тогда
                  // найдём ВИ в ИБ, чтобы определить соответствующий минимальный балл
                  ВступительноеИспытание =
Справочники.ВступительныеИспытания.НайтиПоНаименованию(СтрокаВИ.ВступительноеИспытание)
;
                  МинимальныйБалл = ВступительноеИспытание.МинимальныйБалл;

                  Если Число(СтрокаВИ.Баллы) < Число(МинимальныйБалл) Тогда
                        Сообщение = Новый СообщениеПользователю;
                        Сообщение.Текст = "Не пройден минимальный балл по
вступительному испытанию """ + СтрокаВИ.ВступительноеИспытание + """!";
                        Сообщение.Поле = "ВступительныеИспытания[" +
(СтрокаВИ.НомерСтроки - 1) + "].Баллы";
                        Сообщение.ПутьКДанным = "Объект";
                        Сообщение.Сообщить();

                        Отказ = Истина;
                        Прервать;
                  КонецЕсли;

                  Если Число(СтрокаВИ.Баллы) > 100 Тогда
                        Сообщение = Новый СообщениеПользователю;
                        Сообщение.Текст = "Баллы за вступительное испытание не могут
превышать 100!";
                        Сообщение.Поле = "ВступительныеИспытания[" +
(СтрокаВИ.НомерСтроки - 1) + "].Баллы";
                        Сообщение.ПутьКДанным = "Объект";
                        Сообщение.Сообщить();

                        Отказ = Истина;
                        Прервать;
                  КонецЕсли;
```

Продолжение листинга 2

```
            КонецЕсли;
      КонецЦикла;

      // Проверка того, что указанный год сдачи вступительного испытания не больше
текущего
      Для каждого СтрокаВИ из ВступительныеИспытания Цикл
            Если Год(СтрокаВИ.ГодСдачи) > Год(ТекущаяДата()) Тогда
                  Сообщение = Новый СообщениеПользователю;
                  Сообщение.Текст = "Год сдачи вступительного испытания не может
превышать текущий год!";
                  Сообщение.Поле = "ВступительныеИспытания[" + (СтрокаВИ.НомерСтроки -
1) + "].ГодСдачи";
                  Сообщение.ПутьКДанным = "Объект";
                  Сообщение.Сообщить();

                  Отказ = Истина;
                  Прервать;
            КонецЕсли;
      КонецЦикла;

      // ИД
      Если ИндивидуальныеДостижения.Количество() >= 2 Тогда // во всех последующих
проверках есть смысл, если добавлено 2 и более ИД
            // проверка того, что сумма баллов за индивидуальные достижения не
превышает 10
            СуммаБалловИД = 0;
            // извлекаем каждое добавленное ИД абитуриента из табличной части и
            // ищем его в ИБ, чтобы определить его балльность
            Для каждого Строка из ИндивидуальныеДостижения Цикл
                  Если ЗначениеЗаполнено(Строка.Наименование) Тогда
                        Запрос = Новый Запрос;
                        Запрос.Текст =
                              "ВЫБРАТЬ
                              |     ИндивидуальныеДостижения.Баллы КАК Баллы
                              |ИЗ
                              |     Справочник.ИндивидуальныеДостижения КАК
ИндивидуальныеДостижения
                              |ГДЕ
                              |     ИндивидуальныеДостижения.Наименование =
&Наименование";

                        Запрос.УстановитьПараметр("Наименование",
Строка(Строка.Наименование));

                        РезультатЗапроса = Запрос.Выполнить();

                        ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
                        ВыборкаДетальныеЗаписи.Следующий();

                        СуммаБалловСтроки = ВыборкаДетальныеЗаписи.Баллы;

                        // добавляем найденный балл в общую сумму
                        СуммаБалловИД = СуммаБалловИД + СуммаБалловСтроки;
                  КонецЕсли;
            КонецЦикла;

            Если СуммаБалловИД > 10 Тогда
                  Сообщить("Сумма баллов за индивидуальные достижения не может
превышать 10!");
                  Отказ = Истина;
```

Продолжение листинга 2

```
            КонецЕсли;

            // проверка того, что добавленные индивидуальные достижения не повторяются
            // берём первую строку и сравниваем со всеми последующими, начиная со
второй
            // ...
            // берём m-ю строку и сравниваем с последующими, начиная с (m+1)-й
            Для н = 0 по ИндивидуальныеДостижения.Количество() - 2 Цикл
                  КоличествоПовторяющихсяСтрок = 0;

                  Для к = н + 1 по ИндивидуальныеДостижения.Количество() - 1 Цикл
                        Если ИндивидуальныеДостижения.Получить(н).Наименование =
ИндивидуальныеДостижения.Получить(к).Наименование Тогда
                              КоличествоПовторяющихсяСтрок =
КоличествоПовторяющихсяСтрок + 1;
                        КонецЕсли;
                  КонецЦикла;

                  // если после проверки строки нашлось совпадение - это уже повод
выдать ошибку
                  Если КоличествоПовторяющихсяСтрок > 0 Тогда
                        Сообщить("Добавленные индивидуальные достижения не должны
повторяться!");
                        Отказ = Истина;
                        Прервать;
                  КонецЕсли;
            КонецЦикла;
      КонецЕсли;

      // ОП
      // проверка того, что добавленные образовательные программы не повторяются
      Если ОбразовательныеПрограммы.Количество() >= 2 Тогда // в проверке есть смысл,
если добавлено 2 и более ОП
            // берём первую строку и сравниваем со всеми последующими, начиная со
второй
            // ...
            // берём m-ю строку и сравниваем с последующими, начиная с (m+1)-й
            Для н = 0 по ОбразовательныеПрограммы.Количество() - 2 Цикл
                  КоличествоПовторяющихсяСтрок = 0;

                  Для к = н + 1 по ОбразовательныеПрограммы.Количество() - 1 Цикл
                        Если
ОбразовательныеПрограммы.Получить(н).ОбразовательнаяПрограмма =
ОбразовательныеПрограммы.Получить(к).ОбразовательнаяПрограмма Тогда
                              КоличествоПовторяющихсяСтрок =
КоличествоПовторяющихсяСтрок + 1;
                        КонецЕсли;
                  КонецЦикла;

                  // если после проверки строки нашлось совпадение - это уже повод
выдать ошибку
                  Если КоличествоПовторяющихсяСтрок > 0 Тогда
                        Сообщить("Добавленные образовательные программы не должны
повторяться!");
                        Отказ = Истина;
                        Прервать;
                  КонецЕсли;
            КонецЦикла;
      КонецЕсли;
```

Продолжение листинга 2

```
	// проверка того, что количество добавленных образовательных программ не
превышает допустимое;
	// допустимое значение зависит от того, на какой уровень образования поступает
абитуриент;
	// эти допустимые значения хранятся в соответствующих константах
	Если УровеньОбразованияДляПоступления =
ПредопределенноеЗначение("Перечисление.УровниОбразования.БакалавриатСпециалитет") Тогда
		Если ОбразовательныеПрограммы.Количество() >
Константы.МаксимальноеКоличествоНаправленийБакалавриатаСпециалитета.Получить() Тогда
			Сообщить("Образовательных программ бакалавриата/специалитета для
участия в конкурсе не может быть больше " +
Константы.МаксимальноеКоличествоНаправленийБакалавриатаСпециалитета.Получить() + "!");
			Отказ = Истина;
		КонецЕсли;
	ИначеЕсли УровеньОбразованияДляПоступления =
ПредопределенноеЗначение("Перечисление.УровниОбразования.Магистратура") Тогда
		Если ОбразовательныеПрограммы.Количество() >
Константы.МаксимальноеКоличествоНаправленийМагистратуры.Получить() Тогда
			Сообщить("Образовательных программ магистратуры для участия в
конкурсе не может быть больше " +
Константы.МаксимальноеКоличествоНаправленийМагистратуры.Получить() + "!");
			Отказ = Истина;
		КонецЕсли;
	КонецЕсли;

	// проверка того, что по добавленным образовательным программам внесены все
требуемые вступительные испытания
	Приоритетности = Новый Массив(8);

	// пройдёмся по каждой добавленной образовательной программе абитуриента
	Для каждого СтрокаОП из ОбразовательныеПрограммы Цикл
		Приоритетности[0] = Ложь; // для отметки наличия ВИ 1-го приоритета
		Приоритетности[1] = Ложь; // для отметки наличия ВИ 2-го приоритета
		Приоритетности[2] = Ложь; // ...
		Приоритетности[3] = Ложь; // ...
		Приоритетности[4] = Ложь; // ...
		Приоритетности[5] = Ложь; // ...
		Приоритетности[6] = Ложь; // ...
		Приоритетности[7] = Ложь; // для отметки наличия ВИ 8-го приоритета

		// извлечём из ИБ вступительные испытания, нужные по текущей в цикле
образовательной программе
		Запрос = Новый Запрос;
		Запрос.Текст =
			"ВЫБРАТЬ
			|	ОбразовательныеПрограммы.ВступительныеИспытания.(
			|		ВступительноеИспытание КАК ВступительноеИспытание,
			|		Приоритетность КАК Приоритетность
			|	) КАК ВступительныеИспытания
			|ИЗ
			|	Справочник.ОбразовательныеПрограммы КАК
ОбразовательныеПрограммы
			|ГДЕ
			|	ОбразовательныеПрограммы.Код = &Код";
		Запрос.УстановитьПараметр("Код",
Строка(СтрокаОП.ОбразовательнаяПрограмма));
		РезультатЗапроса = Запрос.Выполнить();
		ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
```

Окончание листинга 2

```
            // пройдёмся по строкам вступительных испытаний образовательных программ и
определим,
            // вступительные испытания каких приоритетов есть в анкете абитуриента (в
табличной части с вступительными испытаниями);
            // если вступительное испытание найдено, записываем в массив приоритетов,
            // что вступительное испытание по такому приоритету есть (этого уже
достаточно в рамках приоритета)
            Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
                  ВступительныеИспытанияОП =
ВыборкаДетальныеЗаписи.ВступительныеИспытания.Выбрать();
                  Пока ВступительныеИспытанияОП.Следующий() Цикл
                        Если Приоритетности[ВступительныеИспытанияОП.Приоритетность -
1] = Ложь Тогда
                              Для каждого СтрокаВИ из ВступительныеИспытания Цикл
                                    Если СтрокаВИ.ВступительноеИспытание =
ВступительныеИспытанияОП.ВступительноеИспытание Тогда
      Приоритетности[ВступительныеИспытанияОП.Приоритетность - 1] = Истина;
                                    КонецЕсли;
                              КонецЦикла;
                        КонецЕсли;
                  КонецЦикла;
            КонецЦикла;

            // пройдёмся ещё раз по строкам вступительных испытаний образовательных
программ абитуриента;
            // если вступительное испытание хотя бы по одному из приоритетов не
обнаружено, выдаём ошибку
            ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
            Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
                  ВступительныеИспытанияОП =
ВыборкаДетальныеЗаписи.ВступительныеИспытания.Выбрать();
                  Пока ВступительныеИспытанияОП.Следующий() Цикл
                        Если Приоритетности[ВступительныеИспытанияОП.Приоритетность -
1] = Ложь Тогда
                              Сообщить("Для образовательной программы """ +
СтрокаОП.ОбразовательнаяПрограмма + """ недостаточно указанных вступительных
испытаний!");
                              Отказ = Истина;
                              Прервать;
                        КонецЕсли;
                  КонецЦикла;
            КонецЦикла;
      КонецЦикла;
КонецПроцедуры
```

Листинг 3 – Программный код модуля формы элемента справочника «Образовательные программы»

```
&НаСервереБезКонтекста
// при выборе пользователем значения "бюджет" автоматически обнуляем стоимость обучения
Процедура
ОснованиеПолученияОбразованияПриИзмененииНаСервере(ОснованиеПолученияОбразования,
СтоимостьОбученияВГод)
      Если ОснованиеПолученияОбразования =
Перечисления.ОснованияПолученияОбразования.Бюджет Тогда
            СтоимостьОбученияВГод = Число("0,00");
      КонецЕсли;
КонецПроцедуры
```

Продолжение листинга 3

```
&НаКлиенте
// вызов на клиенте функции, определённой выше
Процедура ОснованиеПолученияОбразованияПриИзменении(Элемент)
	ОснованиеПолученияОбразованияПриИзмененииНаСервере(Объект.ОснованиеПолученияОбраз
ования, Объект.СтоимостьОбученияВГод);
КонецПроцедуры
```

Листинг 4 – Программный код модуля формы элемента справочника «Документы, удостоверяющие личность»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// меняем маску номера документа в момент открытия формы в зависимости от
гражданства
	Если Объект.Гражданство =
ПредопределенноеЗначение("Справочник.Гражданства.РоссийскаяФедерация") Тогда
		Элементы.Номер.Маска = "9999 999999";
	Иначе
		Элементы.Номер.Маска = "";
	КонецЕсли;

	// определяем доступность функции просмотра файла - скан-копии, если он добавлен
	СканКопия = Объект.СканКопия;
	Элемент = Элементы.Найти("ОткрытьФайл");
	Если ЗначениеЗаполнено(СканКопия) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// при открытии формы уже созданного элемента проверяем,
	// надо ли показывать те или иные скрытые поля
	Если Объект.Гражданство =
ПредопределенноеЗначение("Справочник.Гражданства.РоссийскаяФедерация") Тогда
		Элементы.Номер.Заголовок = "Серия и номер*";
		Элементы.Номер.Маска = "9999 999999";
		Элементы.МестоРождения.Заголовок = "Место рождения*";
		Элементы.КемВыдан.Заголовок = "Кем выдан*";
		Элементы.КодПодразделения.Видимость = Истина;
		Элементы.КодПодразделения.Заголовок = "Код подразделения*";
		Элементы.СрокДействия.Видимость = Ложь;
	ИначеЕсли Объект.Гражданство =
ПредопределенноеЗначение("Справочник.Гражданства.ЛицоБезГражданства") Тогда
		Элементы.Номер.Заголовок = "Номер*";
		Элементы.Номер.Маска = "";
		Элементы.МестоРождения.Заголовок = "Место рождения";
		Элементы.КемВыдан.Заголовок = "Кем выдан";
		Элементы.КодПодразделения.Видимость = Истина;
		Элементы.КодПодразделения.Заголовок = "Код подразделения";
		Элементы.СрокДействия.Видимость = Истина;
		Элементы.СрокДействия.Заголовок = "Срок действия*";
	Иначе
		Элементы.Номер.Заголовок = "Номер*";
		Элементы.Номер.Маска = "";
		Элементы.МестоРождения.Заголовок = "Место рождения*";
		Элементы.КемВыдан.Заголовок = "Кем выдан*";
		Элементы.КодПодразделения.Видимость = Ложь;
		Элементы.СрокДействия.Видимость = Истина;
		Элементы.СрокДействия.Заголовок = "Срок действия*";
```

Продолжение листинга 4

```
	КонецЕсли;

	// Реализация автоматического заполнения полей из формы элемента "Абитуриенты", если текущая форма открыта из неё
	// проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма была открыта из другой управляемой формы
	Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
		// получили форму элемента справочника "Абитуриенты", из которой была открыта текущая форма
		РодительскаяФорма = ЭтаФорма.ВладелецФормы.Родитель.Родитель;

		// далее получаем структуру значений заполнения родительской формы
		// если в текущей форме изначально поле пустое, заполняем его полученными значениями
		Если Не ЗначениеЗаполнено(Объект.ФИОВладельца) Тогда // это означает, что форма "чистая"
			Объект.ФИОВладельца = РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;

			Объект.Гражданство = РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().Гражданство;

			// далее настраиваем видимость и заголовки реквизитов формы в зависимости от формата документа (а формат зависит от гражданства)
			Если РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().Гражданство = ПредопределенноеЗначение("Справочник.Гражданства.РоссийскаяФедерация") Тогда
				Элементы.Номер.Заголовок = "Серия и номер*";
				Элементы.Номер.Маска = "9999 999999";
				ПаспортРФвНаименование("РФ", Объект.Наименование); // записываем в поле реквизита "Наименование" значение "Паспорт гражданина РФ"
				Элементы.МестоРождения.Заголовок = "Место рождения*";
				Элементы.КемВыдан.Заголовок = "Кем выдан*";
				Элементы.КодПодразделения.Видимость = Истина;
				Элементы.КодПодразделения.Заголовок = "Код подразделения*";
				Элементы.СрокДействия.Видимость = Ложь;
			ИначеЕсли РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().Гражданство = ПредопределенноеЗначение("Справочник.Гражданства.ЛицоБезГражданства") Тогда
				Элементы.Номер.Заголовок = "Номер*";
				Элементы.Номер.Маска = "";
				ПаспортРФвНаименование("", Объект.Наименование); // записываем в поле реквизита "Наименование" неопределённое значение
				Элементы.МестоРождения.Заголовок = "Место рождения";
				Элементы.КемВыдан.Заголовок = "Кем выдан";
				Элементы.КодПодразделения.Видимость = Истина;
				Элементы.КодПодразделения.Заголовок = "Код подразделения";
				Элементы.СрокДействия.Видимость = Истина;
				Элементы.СрокДействия.Заголовок = "Срок действия*";
			Иначе
				Элементы.Номер.Заголовок = "Номер*";
				Элементы.Номер.Маска = "";
				ПаспортРФвНаименование("", Объект.Наименование); // записываем в поле реквизита "Наименование" неопределённое значение
				Элементы.МестоРождения.Заголовок = "Место рождения*";
				Элементы.КемВыдан.Заголовок = "Кем выдан*";
				Элементы.КодПодразделения.Видимость = Ложь;
				Элементы.СрокДействия.Видимость = Истина;
				Элементы.СрокДействия.Заголовок = "Срок действия*";
			КонецЕсли;
```

Продолжение листинга 4

```
                    Объект.ПолВладельца =
РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().Пол;
                    Объект.ДатаРождения =
РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ДатаРождения;
            КонецЕсли;
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
// настраиваем видимость и заголовки реквизитов формы в зависимости от формата
документа (а формат зависит от гражданства)
Процедура ГражданствоПриИзменении(Элемент)
      Объект.Код = Неопределено;
      Объект.Наименование = Неопределено;
      Объект.КодПодразделения = Неопределено;
      Объект.СрокДействия = Неопределено;

      Если Объект.Гражданство =
ПредопределенноеЗначение("Справочник.Гражданства.РоссийскаяФедерация") Тогда
            Элементы.Номер.Заголовок = "Серия и номер*";
            Элементы.Номер.Маска = "9999 999999";
            ПаспортРФвНаименование("РФ", Объект.Наименование); // записываем в поле
реквизита "Наименование" значение "Паспорт гражданина РФ"
            Элементы.МестоРождения.Заголовок = "Место рождения*";
            Элементы.КемВыдан.Заголовок = "Кем выдан*";
            Элементы.КодПодразделения.Видимость = Истина;
            Элементы.КодПодразделения.Заголовок = "Код подразделения*";
            Элементы.СрокДействия.Видимость = Ложь;
      ИначеЕсли Объект.Гражданство =
ПредопределенноеЗначение("Справочник.Гражданства.ЛицоБезГражданства") Тогда
            Элементы.Номер.Заголовок = "Номер*";
            Элементы.Номер.Маска = "";
            ПаспортРФвНаименование("", Объект.Наименование); // записываем в поле
реквизита "Наименование" неопределённое значение
            Элементы.МестоРождения.Заголовок = "Место рождения";
            Элементы.КемВыдан.Заголовок = "Кем выдан";
            Элементы.КодПодразделения.Видимость = Истина;
            Элементы.КодПодразделения.Заголовок = "Код подразделения";
            Элементы.СрокДействия.Видимость = Истина;
            Элементы.СрокДействия.Заголовок = "Срок действия*";
      Иначе
            Элементы.Номер.Заголовок = "Номер*";
            Элементы.Номер.Маска = "";
            ПаспортРФвНаименование("", Объект.Наименование); // записываем в поле
реквизита "Наименование" неопределённое значение
            Элементы.МестоРождения.Заголовок = "Место рождения*";
            Элементы.КемВыдан.Заголовок = "Кем выдан*";
            Элементы.КодПодразделения.Видимость = Ложь;
            Элементы.СрокДействия.Видимость = Истина;
            Элементы.СрокДействия.Заголовок = "Срок действия*";
      КонецЕсли;
КонецПроцедуры

&НаСервереБезКонтекста
Процедура ПаспортРФвНаименование(Признак, Наименование)
      Если Признак = "РФ" Тогда
            Наименование = "Паспорт гражданина Российской Федерации";
      Иначе
            Наименование = Неопределено;
      КонецЕсли;
```

Продолжение листинга 4

```
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаРожденияПриИзменении(Элемент)
	Если Объект.ДатаРождения > ТекущаяДата() Тогда
		Предупреждение("Нельзя указать ещё не наступившую дату!");
		Объект.ДатаРождения = Неопределено;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаВыдачиПриИзменении(Элемент)
	Если Объект.ДатаВыдачи > ТекущаяДата() Тогда
		Предупреждение("Нельзя указать ещё не наступившую дату!");
		Объект.ДатаВыдачи = Неопределено;
	КонецЕсли;
КонецПроцедуры

&НаСервере
// вызов процедуры проверки заполнения из модуля объекта
Процедура ПередЗаписьюНаСервере(Отказ, ТекущийОбъект, ПараметрыЗаписи)
	Документ = РеквизитФормыВЗначение("Объект");
	Документ.ПроверитьЗаполнение();
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-
копия документа
	СтандартнаяОбработка = Ложь;
	Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
	Диалог.Заголовок = "Выберите файл";
	Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
	Диалог.МножественныйВыбор = Ложь;

	// если файл уже добавлен, получаем его путь и открываем Проводник по нему
	Если Не Объект.СканКопия = Неопределено Тогда
		ФайлPDF = Новый Файл(Объект.СканКопия);
		Если ФайлPDF.Существует() Тогда
			Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
			Диалог.Каталог = ФайлPDF.Путь;
		КонецЕсли;
	Иначе // открываем Проводник в корне диска C
		Диалог.ПолноеИмяФайла = "";
		Диалог.Каталог = "C:\";
	КонецЕсли;

	// если файл был выбран
	Если Диалог.Выбрать() Тогда
		Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
		Элемент = Элементы.Найти("ОткрытьФайл");
		Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия документа
```

Окончание листинга 4

```
	ФайлPDF = Новый Файл(Объект.СканКопия);
	Если ФайлPDF.Существует() Тогда
		ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
	Иначе
		Сообщение = Новый СообщениеПользователю;
		Сообщение.Текст = "Файл не существует!";
		Сообщение.Поле = "СканКопия";
		Сообщение.ПутьКДанным = "Объект";
		Сообщение.Сообщить();
	КонецЕсли;
КонецПроцедуры
```

Листинг 5 – Программный код модуля объекта справочника «Документы, удостоверяющие личность»

```
Процедура ОбработкаПроверкиЗаполнения(Отказ, ПроверяемыеРеквизиты) Экспорт
	// т. к. в зависимости от гражданства меняется форма документа, удостоверяющего личность,
	// необходимо менять необходимость заполнения определённых полей при выборе гражданства
	Если Гражданство <> Справочники.Гражданства.ЛицоБезГражданства Тогда
		ПроверяемыеРеквизиты.Добавить("МестоРождения");
	КонецЕсли;

	Если Гражданство = Справочники.Гражданства.РоссийскаяФедерация Тогда
		ПроверяемыеРеквизиты.Добавить("КодПодразделения");
	Иначе
		ПроверяемыеРеквизиты.Добавить("СрокДействия");
	КонецЕсли;
КонецПроцедуры
```

Листинг 6 – Программный код модуля формы элемента справочника «Подтверждающие документы иностранных граждан и лиц без гражданства»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// определяем доступность функции просмотра файла - скан-копии, если он добавлен
	СканКопия = Объект.СканКопия;
	Элемент = Элементы.Найти("ОткрытьФайл");
	Если ЗначениеЗаполнено(СканКопия) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// Реализация автоматического заполнения полей из формы элемента "Абитуриенты", если текущая форма открыта из неё
	// проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма была открыта из другой управляемой формы
	Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
		// получили форму элемента справочника "Абитуриенты", из которой была открыта текущая форма
		РодительскаяФорма = ЭтаФорма.ВладелецФормы.Родитель.Родитель.Родитель;

		// далее получаем структуру значений заполнения родительской формы
		// если в текущей форме изначально поле пустое, заполняем его полученными значениями
		Если Не ЗначениеЗаполнено(Объект.ФИОАбитуриента) Тогда
```

Продолжение листинга 6

```
                    Объект.ФИОАбитуриента =
РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;
            КонецЕсли;
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаВыдачиПриИзменении(Элемент)
      Если Объект.ДатаВыдачи > ТекущаяДата() Тогда
            Предупреждение("Нельзя указать ещё не наступившую дату!");
            Объект.ДатаВыдачи = Неопределено;
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-
копия документа
      СтандартнаяОбработка = Ложь;
      Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
      Диалог.Заголовок = "Выберите файл";
      Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
      Диалог.МножественныйВыбор = Ложь;

      // если файл уже добавлен, получаем его путь и открываем Проводник по нему
      Если Не Объект.СканКопия = Неопределено Тогда
            ФайлPDF = Новый Файл(Объект.СканКопия);
            Если ФайлPDF.Существует() Тогда
                  Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
                  Диалог.Каталог = ФайлPDF.Путь;
            КонецЕсли;
      Иначе // открываем Проводник в корне диска С
            Диалог.ПолноеИмяФайла = "";
            Диалог.Каталог = "C:\";
      КонецЕсли;

      // если файл был выбран
      Если Диалог.Выбрать() Тогда
            Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
            Элемент = Элементы.Найти("ОткрытьФайл");
            Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия документа
      ФайлPDF = Новый Файл(Объект.СканКопия);
      Если ФайлPDF.Существует() Тогда
            ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
      Иначе
            Сообщение = Новый СообщениеПользователю;
            Сообщение.Текст = "Файл не существует!";
            Сообщение.Поле = "СканКопия";
            Сообщение.ПутьКДанным = "Объект";
            Сообщение.Сообщить();
      КонецЕсли;
КонецПроцедуры
```

Листинг 7 – Программный код модуля формы элемента справочника «СНИЛС»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// определяем доступность функции просмотра файла - скан-копии, если он добавлен
	СканКопия = Объект.СканКопия;
	Элемент = Элементы.Найти("ОткрытьФайл");
	Если ЗначениеЗаполнено(СканКопия) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// Реализация автоматического заполнения полей из формы элемента "Абитуриенты", если текущая форма открыта из неё
	// проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма была открыта из другой управляемой формы
	Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
		// получили форму элемента справочника "Абитуриенты", из которой была открыта текущая форма
		РодительскаяФорма = ЭтаФорма.ВладелецФормы.Родитель.Родитель;

		// далее получаем структуру значений заполнения родительской формы
		// если в текущей форме изначально поле пустое, заполняем его полученными значениями
		Если Не ЗначениеЗаполнено(Объект.Наименование) Тогда // это означает, что форма "чистая"
			Объект.Наименование = РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;
			Объект.ПолВладельца = РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().Пол;
			Объект.ДатаРождения = РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ДатаРождения;
		КонецЕсли;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаРожденияПриИзменении(Элемент)
	Если Объект.ДатаРождения > ТекущаяДата() Тогда
		Предупреждение("Нельзя указать ещё не наступившую дату!");
		Объект.ДатаРождения = Неопределено;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаРегистрацииПриИзменении(Элемент)
	Если Объект.ДатаРегистрации > ТекущаяДата() Тогда
		Предупреждение("Нельзя указать ещё не наступившую дату!");
		Объект.ДатаРегистрации = Неопределено;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-копия СНИЛС
	СтандартнаяОбработка = Ложь;
	Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
	Диалог.Заголовок = "Выберите файл";
	Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
```

Продолжение листинга 7

```
	Диалог.МножественныйВыбор = Ложь;

	// если файл уже добавлен, получаем его путь и открываем Проводник по нему
	Если Не Объект.СканКопия = Неопределено Тогда
		ФайлPDF = Новый Файл(Объект.СканКопия);
		Если ФайлPDF.Существует() Тогда
			Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
			Диалог.Каталог = ФайлPDF.Путь;
		КонецЕсли;
	Иначе // открываем Проводник в корне диска С
		Диалог.ПолноеИмяФайла = "";
		Диалог.Каталог = "C:\";
	КонецЕсли;

	// если файл был выбран
	Если Диалог.Выбрать() Тогда
		Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
		Элемент = Элементы.Найти("ОткрытьФайл");
		Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия СНИЛС
	ФайлPDF = Новый Файл(Объект.СканКопия);
	Если ФайлPDF.Существует() Тогда
		ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
	Иначе
		Сообщение = Новый СообщениеПользователю;
		Сообщение.Текст = "Файл не существует!";
		Сообщение.Поле = "СканКопия";
		Сообщение.ПутьКДанным = "Объект";
		Сообщение.Сообщить();
	КонецЕсли;
КонецПроцедуры
```

Листинг 8 – Программный код модуля формы элемента справочника «Документы об образовании»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// определяем доступность функции просмотра файла - скан-копии, если он добавлен
	СканКопия = Объект.СканКопия;
	Элемент = Элементы.Найти("ОткрытьФайл");
	Если ЗначениеЗаполнено(СканКопия) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// Реализация автоматического заполнения полей из формы элемента "Абитуриенты",
если текущая форма открыта из неё
	// проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма
была открыта из другой управляемой формы
	Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
		// получили форму элемента справочника "Абитуриенты", из которой была
открыта текущая форма
		РодительскаяФорма = ЭтаФорма.ВладелецФормы.Родитель.Родитель.Родитель;
```

Продолжение листинга 8

```
            // далее получаем структуру значений заполнения родительской формы
            // если в текущей форме изначально поле пустое, заполняем его полученными
значениями
            Если Не ЗначениеЗаполнено(Объект.ФИОВладельца) Тогда
                  Объект.ФИОВладельца =
РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;
            КонецЕсли;
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаВыдачиПриИзменении(Элемент)
      Если Объект.ДатаВыдачи > ТекущаяДата() Тогда
            Предупреждение("Нельзя указать ещё не наступившую дату!");
            Объект.ДатаВыдачи = Неопределено;
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-
копия документа
      СтандартнаяОбработка = Ложь;
      Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
      Диалог.Заголовок = "Выберите файл";
      Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
      Диалог.МножественныйВыбор = Ложь;

      // если файл уже добавлен, получаем его путь и открываем Проводник по нему
      Если Не Объект.СканКопия = Неопределено Тогда
            ФайлPDF = Новый Файл(Объект.СканКопия);
            Если ФайлPDF.Существует() Тогда
                  Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
                  Диалог.Каталог = ФайлPDF.Путь;
            КонецЕсли;
      Иначе // открываем Проводник в корне диска С
            Диалог.ПолноеИмяФайла = "";
            Диалог.Каталог = "C:\";
      КонецЕсли;

      // если файл был выбран
      Если Диалог.Выбрать() Тогда
            Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
            Элемент = Элементы.Найти("ОткрытьФайл");
            Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия документа
      ФайлPDF = Новый Файл(Объект.СканКопия);
      Если ФайлPDF.Существует() Тогда
            ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
      Иначе
            Сообщение = Новый СообщениеПользователю;
            Сообщение.Текст = "Файл не существует!";
            Сообщение.Поле = "СканКопия";
            Сообщение.ПутьКДанным = "Объект";
```

Окончание листинга 8

```
            Сообщение.Сообщить();
      КонецЕсли;
КонецПроцедуры
```

Листинг 9 – Программный код модуля формы элемента справочника «Документы, подтверждающие наличие индивидуального достижения»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
      // определяем доступность функции просмотра файла - скан-копии, если он добавлен
      СканКопия = Объект.СканКопия;
      Элемент = Элементы.Найти("ОткрытьФайл");
      Если ЗначениеЗаполнено(СканКопия) Тогда
            Элемент.Доступность = Истина;
      Иначе
            Элемент.Доступность = Ложь;
      КонецЕсли;

      // Реализация автоматического заполнения полей из формы элемента "Абитуриенты",
если текущая форма открыта из неё
      // проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма
была открыта из другой управляемой формы
      Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
            // получили форму элемента справочника "Абитуриенты", из которой была
открыта текущая форма
            РодительскаяФорма =
ЭтаФорма.ВладелецФормы.Родитель.Родитель.Родитель.Родитель;

            // далее получаем структуру значений заполнения родительской формы
            // если в текущей форме изначально поле пустое, заполняем его полученными
значениями
            Если Не ЗначениеЗаполнено(Объект.ФИОАбитуриента) Тогда // это означает,
что форма "чистая"
                  Объект.ФИОАбитуриента =
РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;
                  Объект.ИндивидуальноеДостижение =
РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ИндивидуальноеДостижение;
            КонецЕсли;
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаВыдачиПриИзменении(Элемент)
      Если Объект.ДатаВыдачи > ТекущаяДата() Тогда
            Предупреждение("Нельзя указать ещё не наступившую дату!");
            Объект.ДатаВыдачи = Неопределено;
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-
копия документа
      СтандартнаяОбработка = Ложь;
      Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
      Диалог.Заголовок = "Выберите файл";
      Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
      Диалог.МножественныйВыбор = Ложь;
```

Продолжение листинга 9

```
	// если файл уже добавлен, получаем его путь и открываем Проводник по нему
	Если Не Объект.СканКопия = Неопределено Тогда
		ФайлPDF = Новый Файл(Объект.СканКопия);
		Если ФайлPDF.Существует() Тогда
			Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
			Диалог.Каталог = ФайлPDF.Путь;
		КонецЕсли;
	Иначе // открываем Проводник в корне диска C
		Диалог.ПолноеИмяФайла = "";
		Диалог.Каталог = "C:\";
	КонецЕсли;

	// если файл был выбран
	Если Диалог.Выбрать() Тогда
		Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
		Элемент = Элементы.Найти("ОткрытьФайл");
		Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия документа
	ФайлPDF = Новый Файл(Объект.СканКопия);
	Если ФайлPDF.Существует() Тогда
		ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
	Иначе
		Сообщение = Новый СообщениеПользователю;
		Сообщение.Текст = "Файл не существует!";
		Сообщение.Поле = "СканКопия";
		Сообщение.ПутьКДанным = "Объект";
		Сообщение.Сообщить();
	КонецЕсли;
КонецПроцедуры
```

Листинг 10 – Программный код модуля формы элемента справочника «Документы, подтверждающие преимущественное право»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// определяем доступность функции просмотра файла - скан-копии, если он добавлен
	СканКопия = Объект.СканКопия;
	Элемент = Элементы.Найти("ОткрытьФайл");
	Если ЗначениеЗаполнено(СканКопия) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// Реализация автоматического заполнения полей из формы элемента "Абитуриенты",
если текущая форма открыта из неё
	// проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма
была открыта из другой управляемой формы
	Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
		// получили форму элемента справочника "Абитуриенты", из которой была
открыта текущая форма
		РодительскаяФорма = ЭтаФорма.ВладелецФормы.Родитель.Родитель;

		// далее получаем структуру значений заполнения родительской формы
```

Продолжение листинга 10

```
            // если в текущей форме изначально поле пустое, заполняем его полученными
значениями
            Если Не ЗначениеЗаполнено(Объект.ФИОАбитуриента) Тогда
                  Объект.ФИОАбитуриента =
РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;
            КонецЕсли;
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаВыдачиПриИзменении(Элемент)
      Если Объект.ДатаВыдачи > ТекущаяДата() Тогда
            Предупреждение("Нельзя указать ещё не наступившую дату!");
            Объект.ДатаВыдачи = Неопределено;
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-
копия документа
      СтандартнаяОбработка = Ложь;
      Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
      Диалог.Заголовок = "Выберите файл";
      Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
      Диалог.МножественныйВыбор = Ложь;

      // если файл уже добавлен, получаем его путь и открываем Проводник по нему
      Если Не Объект.СканКопия = Неопределено Тогда
            ФайлPDF = Новый Файл(Объект.СканКопия);
            Если ФайлPDF.Существует() Тогда
                  Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
                  Диалог.Каталог = ФайлPDF.Путь;
            КонецЕсли;
      Иначе // открываем Проводник в корне диска С
            Диалог.ПолноеИмяФайла = "";
            Диалог.Каталог = "C:\";
      КонецЕсли;

      Если Диалог.Выбрать() Тогда // если файл был выбран
            Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
            Элемент = Элементы.Найти("ОткрытьФайл");
            Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия документа
      ФайлPDF = Новый Файл(Объект.СканКопия);
      Если ФайлPDF.Существует() Тогда ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
      Иначе
            Сообщение = Новый СообщениеПользователю;
            Сообщение.Текст = "Файл не существует!";
            Сообщение.Поле = "СканКопия";
            Сообщение.ПутьКДанным = "Объект";
            Сообщение.Сообщить();
      КонецЕсли;
КонецПроцедуры
```

Листинг 11 – Программный код модуля формы элемента справочника «Документы, подтверждающие особое право»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// определяем доступность функции просмотра файла - скан-копии, если он добавлен
	СканКопия = Объект.СканКопия;
	Элемент = Элементы.Найти("ОткрытьФайл");
	Если ЗначениеЗаполнено(СканКопия) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// Реализация автоматического заполнения полей из формы элемента "Абитуриенты", если текущая форма открыта из неё
	// проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма была открыта из другой управляемой формы
	Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
		// получили форму элемента справочника "Абитуриенты", из которой была открыта текущая форма
		РодительскаяФорма = ЭтаФорма.ВладелецФормы.Родитель.Родитель;

		// далее получаем структуру значений заполнения родительской формы
		// если в текущей форме изначально поле пустое, заполняем его полученными значениями
		Если Не ЗначениеЗаполнено(Объект.ФИОАбитуриента) Тогда
			Объект.ФИОАбитуриента = РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;
		КонецЕсли;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаВыдачиПриИзменении(Элемент)
	Если Объект.ДатаВыдачи > ТекущаяДата() Тогда
		Предупреждение("Нельзя указать ещё не наступившую дату!");
		Объект.ДатаВыдачи = Неопределено;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-копия документа
	СтандартнаяОбработка = Ложь;
	Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
	Диалог.Заголовок = "Выберите файл";
	Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
	Диалог.МножественныйВыбор = Ложь;

	// если файл уже добавлен, получаем его путь и открываем Проводник по нему
	Если Не Объект.СканКопия = Неопределено Тогда
		ФайлPDF = Новый Файл(Объект.СканКопия);
		Если ФайлPDF.Существует() Тогда
			Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
			Диалог.Каталог = ФайлPDF.Путь;
		КонецЕсли;
	Иначе // открываем Проводник в корне диска С
		Диалог.ПолноеИмяФайла = "";
		Диалог.Каталог = "C:\";
```

Продолжение листинга 11

```
	КонецЕсли;

	// если файл был выбран
	Если Диалог.Выбрать() Тогда
		Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
		Элемент = Элементы.Найти("ОткрытьФайл");
		Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия документа
	ФайлPDF = Новый Файл(Объект.СканКопия);
	Если ФайлPDF.Существует() Тогда
		ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
	Иначе
		Сообщение = Новый СообщениеПользователю;
		Сообщение.Текст = "Файл не существует!";
		Сообщение.Поле = "СканКопия";
		Сообщение.ПутьКДанным = "Объект";
		Сообщение.Сообщить();
	КонецЕсли;
КонецПроцедуры
```

Листинг 12 – Программный код модуля формы элемента справочника «Договоры о целевом обучении»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// определяем доступность функции просмотра файла - скан-копии, если он добавлен
	СканКопия = Объект.СканКопия;
	Элемент = Элементы.Найти("ОткрытьФайл");
	Если ЗначениеЗаполнено(СканКопия) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// Реализация автоматического заполнения полей из формы элемента "Абитуриенты",
если текущая форма открыта из неё
	// проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма
была открыта из другой управляемой формы
	Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
		// получили форму элемента справочника "Абитуриенты", из которой была
открыта текущая форма
		РодительскаяФорма = ЭтаФорма.ВладелецФормы.Родитель.Родитель;

		// далее получаем структуру значений заполнения родительской формы
		// если в текущей форме изначально поле пустое, заполняем его полученными
значениями
		Если Не ЗначениеЗаполнено(Объект.ФИОАбитуриента) Тогда
			Объект.ФИОАбитуриента =
РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;
		КонецЕсли;
	КонецЕсли;
КонецПроцедуры
```

Продолжение листинга 12

```
&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаЗаключенияПриИзменении(Элемент)
	Если Объект.ДатаЗаключения > ТекущаяДата() Тогда
		Предупреждение("Нельзя указать ещё не наступившую дату!");
		Объект.ДатаЗаключения = Неопределено;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-
копия документа
	СтандартнаяОбработка = Ложь;
	Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
	Диалог.Заголовок = "Выберите файл";
	Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
	Диалог.МножественныйВыбор = Ложь;

	// если файл уже добавлен, получаем его путь и открываем Проводник по нему
	Если Не Объект.СканКопия = Неопределено Тогда
		ФайлPDF = Новый Файл(Объект.СканКопия);
		Если ФайлPDF.Существует() Тогда
			Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
			Диалог.Каталог = ФайлPDF.Путь;
		КонецЕсли;
	Иначе // открываем Проводник в корне диска C
		Диалог.ПолноеИмяФайла = "";
		Диалог.Каталог = "C:\";
	КонецЕсли;

	// если файл был выбран
	Если Диалог.Выбрать() Тогда
		Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
		Элемент = Элементы.Найти("ОткрытьФайл");
		Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия документа
	ФайлPDF = Новый Файл(Объект.СканКопия);
	Если ФайлPDF.Существует() Тогда
		ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
	Иначе
		Сообщение = Новый СообщениеПользователю;
		Сообщение.Текст = "Файл не существует!";
		Сообщение.Поле = "СканКопия";
		Сообщение.ПутьКДанным = "Объект";
		Сообщение.Сообщить();
	КонецЕсли;
КонецПроцедуры
```

Листинг 13 – Программный код модуля формы элемента справочника «Документы, подтверждающие статус победителя или призёра олимпиад или спортивных соревнований высокого уровня»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// определяем доступность функции просмотра файла - скан-копии, если он добавлен
	СканКопия = Объект.СканКопия;
	Элемент = Элементы.Найти("ОткрытьФайл");
	Если ЗначениеЗаполнено(СканКопия) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// Реализация автоматического заполнения полей из формы элемента "Абитуриенты", если текущая форма открыта из неё
	// проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма была открыта из другой управляемой формы
	Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
		// получили форму элемента справочника "Абитуриенты", из которой была открыта текущая форма
		РодительскаяФорма = ЭтаФорма.ВладелецФормы.Родитель.Родитель;

		// далее получаем структуру значений заполнения родительской формы
		// если в текущей форме изначально поле пустое, заполняем его полученными значениями
		Если Не ЗначениеЗаполнено(Объект.ФИОАбитуриента) Тогда
			Объект.ФИОАбитуриента = РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;
		КонецЕсли;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаВыдачиПриИзменении(Элемент)
	Если Объект.ДатаВыдачи > ТекущаяДата() Тогда
		Предупреждение("Нельзя указать ещё не наступившую дату!");
		Объект.ДатаВыдачи = Неопределено;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-копия документа
	СтандартнаяОбработка = Ложь;
	Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
	Диалог.Заголовок = "Выберите файл";
	Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
	Диалог.МножественныйВыбор = Ложь;

	// если файл уже добавлен, получаем его путь и открываем Проводник по нему
	Если Не Объект.СканКопия = Неопределено Тогда
		ФайлPDF = Новый Файл(Объект.СканКопия);
		Если ФайлPDF.Существует() Тогда
			Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
			Диалог.Каталог = ФайлPDF.Путь;
		КонецЕсли;
	Иначе // открываем Проводник в корне диска C
```

Продолжение листинга 13

```
            Диалог.ПолноеИмяФайла = "";
            Диалог.Каталог = "C:\";
      КонецЕсли;

      // если файл был выбран
      Если Диалог.Выбрать() Тогда
            Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
            Элемент = Элементы.Найти("ОткрытьФайл");
            Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия документа
      ФайлPDF = Новый Файл(Объект.СканКопия);
      Если ФайлPDF.Существует() Тогда
            ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
      Иначе
            Сообщение = Новый СообщениеПользователю;
            Сообщение.Текст = "Файл не существует!";
            Сообщение.Поле = "СканКопия";
            Сообщение.ПутьКДанным = "Объект";
            Сообщение.Сообщить();
      КонецЕсли;
КонецПроцедуры
```

Листинг 14 – Программный код модуля формы документа документа «Заявление на участие в конкурсе»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
      // определяем доступность функции просмотра файла - скан-копии, если он добавлен
      СканКопия = Объект.СканКопия;
      Элемент = Элементы.Найти("ОткрытьФайл");
      Если ЗначениеЗаполнено(СканКопия) Тогда
            Элемент.Доступность = Истина;
      Иначе
            Элемент.Доступность = Ложь;
      КонецЕсли;

      // Реализация автоматического заполнения полей из формы элемента "Абитуриенты",
если текущая форма открыта из неё
      // проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма
была открыта из другой управляемой формы
      Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
            // получили форму элемента справочника "Абитуриенты", из которой была
открыта текущая форма
            РодительскаяФорма = ЭтаФорма.ВладелецФормы.Родитель.Родитель;

            // далее получаем структуру значений заполнения родительской формы
            // если в текущей форме изначально поле пустое, заполняем его полученными
значениями
            Если Не ЗначениеЗаполнено(Объект.ФИОАбитуриента) Тогда
                  Объект.ФИОАбитуриента =
РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;
            КонецЕсли;
      КонецЕсли;
КонецПроцедуры
```

Продолжение листинга 14

```
&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаПриИзменении(Элемент)
	Если Объект.Дата > ТекущаяДата() Тогда
		Предупреждение("Нельзя указать ещё не наступившую дату!");
		Объект.Дата = Неопределено;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-
копия заявления на участие в конкурсе
	СтандартнаяОбработка = Ложь;
	Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
	Диалог.Заголовок = "Выберите файл";
	Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
	Диалог.МножественныйВыбор = Ложь;

	// если файл уже добавлен, получаем его путь и открываем Проводник по нему
	Если Не Объект.СканКопия = Неопределено Тогда
		ФайлPDF = Новый Файл(Объект.СканКопия);
		Если ФайлPDF.Существует() Тогда
			Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
			Диалог.Каталог = ФайлPDF.Путь;
		КонецЕсли;
	Иначе // открываем Проводник в корне диска C
		Диалог.ПолноеИмяФайла = "";
		Диалог.Каталог = "C:\";
	КонецЕсли;

	// если файл был выбран
	Если Диалог.Выбрать() Тогда
		Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
		Элемент = Элементы.Найти("ОткрытьФайл");
		Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия заявления на участие в конкурсе
	ФайлPDF = Новый Файл(Объект.СканКопия);
	Если ФайлPDF.Существует() Тогда
		ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
	Иначе
		Сообщение = Новый СообщениеПользователю;
		Сообщение.Текст = "Файл не существует!";
		Сообщение.Поле = "СканКопия";
		Сообщение.ПутьКДанным = "Объект";
		Сообщение.Сообщить();
	КонецЕсли;
КонецПроцедуры
```

Листинг 15 – Программный код модуля формы документа документа «Согласие на обработку персональных данных»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// определяем доступность функции просмотра файла - скан-копии, если он добавлен
```

Продолжение листинга 15

```
	СканКопия = Объект.СканКопия;
	Элемент = Элементы.Найти("ОткрытьФайл");
	Если ЗначениеЗаполнено(СканКопия) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// Реализация автоматического заполнения полей из формы элемента "Абитуриенты", 
если текущая форма открыта из неё
	// проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма 
была открыта из другой управляемой формы
	Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
		// получили форму элемента справочника "Абитуриенты", из которой была 
открыта текущая форма
		РодительскаяФорма = ЭтаФорма.ВладелецФормы.Родитель.Родитель;

		// далее получаем структуру значений заполнения родительской формы
		// если в текущей форме изначально поле пустое, заполняем его полученными 
значениями
		Если Не ЗначениеЗаполнено(Объект.ФИОАбитуриента) Тогда
			Объект.ФИОАбитуриента = 
РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;
		КонецЕсли;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
// проверка того, что количество добавленных образовательных программ не превышает 
допустимое
// допустимое значение зависит от того, на какой уровень образования поступает 
абитуриент
// эти допустимые значения хранятся в соответствующих константах
Процедура ДатаПриИзменении(Элемент)
	Если Объект.Дата > ТекущаяДата() Тогда
		Предупреждение("Нельзя указать ещё не наступившую дату!");
		Объект.Дата = Неопределено;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-
копия согласия на обработку п/д
	СтандартнаяОбработка = Ложь;
	Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
	Диалог.Заголовок = "Выберите файл";
	Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
	Диалог.МножественныйВыбор = Ложь;

	// если файл уже добавлен, получаем его путь и открываем Проводник по нему
	Если Не Объект.СканКопия = Неопределено Тогда
		ФайлPDF = Новый Файл(Объект.СканКопия);
		Если ФайлPDF.Существует() Тогда
			Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
			Диалог.Каталог = ФайлPDF.Путь;
		КонецЕсли;
	Иначе // открываем Проводник в корне диска С
		Диалог.ПолноеИмяФайла = "";
		Диалог.Каталог = "C:\";
	КонецЕсли;
```

Окончание листинга 15

```
	// если файл был выбран
	Если Диалог.Выбрать() Тогда
		Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
		Элемент = Элементы.Найти("ОткрытьФайл");
		Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия согласия на обработку п/д
	ФайлPDF = Новый Файл(Объект.СканКопия);
	Если ФайлPDF.Существует() Тогда
		ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
	Иначе
		Сообщение = Новый СообщениеПользователю;
		Сообщение.Текст = "Файл не существует!";
		Сообщение.Поле = "СканКопия";
		Сообщение.ПутьКДанным = "Объект";
		Сообщение.Сообщить();
	КонецЕсли;
КонецПроцедуры
```

Листинг 16 – Программный код модуля формы документа документа «Согласие о зачислении»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// определяем доступность функции просмотра файла - скан-копии, если он добавлен
	СканКопия = Объект.СканКопия;
	Элемент = Элементы.Найти("ОткрытьФайл");
	Если ЗначениеЗаполнено(СканКопия) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// Реализация автоматического заполнения полей из формы элемента "Абитуриенты",
если текущая форма открыта из неё
	// проверяем, откуда была открыта форма: если тип владельца - ПолеФормы, то форма
была открыта из другой управляемой формы
	Если ТипЗнч(ЭтаФорма.ВладелецФормы) = Тип("ПолеФормы") Тогда
		// получили форму элемента справочника "Абитуриенты", из которой была
открыта текущая форма
		РодительскаяФорма = ЭтаФорма.ВладелецФормы.Родитель.Родитель;

		// далее получаем структуру значений заполнения родительской формы
		// если в текущей форме изначально поле пустое, заполняем его полученными
значениями
		Если Не ЗначениеЗаполнено(Объект.ФИОАбитуриента) Тогда
			Объект.ФИОАбитуриента =
РодительскаяФорма.ПередачаЗначенийВФормыЭлемента().ФИО;
		КонецЕсли;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаПриИзменении(Элемент)
	Если Объект.Дата > ТекущаяДата() Тогда
		Предупреждение("Нельзя указать ещё не наступившую дату!");
```

Продолжение листинга 16

```
            Объект.Дата = Неопределено;
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-
копия согласия о зачислении
      СтандартнаяОбработка = Ложь;
      Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
      Диалог.Заголовок = "Выберите файл";
      Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
      Диалог.МножественныйВыбор = Ложь;

      // если файл уже добавлен, получаем его путь и открываем Проводник по нему
      Если Не Объект.СканКопия = Неопределено Тогда
            ФайлPDF = Новый Файл(Объект.СканКопия);
            Если ФайлPDF.Существует() Тогда
                  Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
                  Диалог.Каталог = ФайлPDF.Путь;
            КонецЕсли;
      Иначе // открываем Проводник в корне диска С
            Диалог.ПолноеИмяФайла = "";
            Диалог.Каталог = "С:\";
      КонецЕсли;

      // если файл был выбран
      Если Диалог.Выбрать() Тогда
            Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
            Элемент = Элементы.Найти("ОткрытьФайл");
            Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия согласия о зачислении
      ФайлPDF = Новый Файл(Объект.СканКопия);
      Если ФайлPDF.Существует() Тогда
            ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
      Иначе
            Сообщение = Новый СообщениеПользователю;
            Сообщение.Текст = "Файл не существует!";
            Сообщение.Поле = "СканКопия";
            Сообщение.ПутьКДанным = "Объект";
            Сообщение.Сообщить();
      КонецЕсли;
КонецПроцедуры
```

Листинг 17 – Программный код модуля формы документа документа «Приказ о зачислении»

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
      // определяем доступность функции просмотра файла - скан-копии, если он добавлен
      СканКопия = Объект.СканКопия;
      Элемент = Элементы.Найти("ОткрытьФайл");
      Если ЗначениеЗаполнено(СканКопия) Тогда
            Элемент.Доступность = Истина;
      Иначе
            Элемент.Доступность = Ложь;
      КонецЕсли;
```

Продолжение листинга 17

```
	// определяем доступность функции формирования списка абитуриентов
	ОбразовательнаяПрограмма = Объект.ОбразовательнаяПрограмма;
	Элемент = Элементы.Найти("СформироватьСписокАбитуриентов");
	Если ЗначениеЗаполнено(ОбразовательнаяПрограмма) Тогда
		Элемент.Доступность = Истина;
	Иначе
		Элемент.Доступность = Ложь;
	КонецЕсли;

	// если при открытии формы документа (уже созданного) ОП заполнена
	Если ЗначениеЗаполнено(ОбразовательнаяПрограмма) Тогда
		// обратимся к серверу и получим максимальную приоритетность ВИ по ОП
		МаксимальнаяПриоритетность =
ПолучитьМаксимальнуюПриоритетность(ОбразовательнаяПрограмма);

		// сначала скроем все столбцы с ВИ
		Элементы.АбитуриентыБаллыЗаВИ1.Видимость = Ложь;
		Элементы.АбитуриентыБаллыЗаВИ2.Видимость = Ложь;
		Элементы.АбитуриентыБаллыЗаВИ3.Видимость = Ложь;
		Элементы.АбитуриентыБаллыЗаВИ4.Видимость = Ложь;
		Элементы.АбитуриентыБаллыЗаВИ5.Видимость = Ложь;
		Элементы.АбитуриентыБаллыЗаВИ6.Видимость = Ложь;
		Элементы.АбитуриентыБаллыЗаВИ7.Видимость = Ложь;
		Элементы.АбитуриентыБаллыЗаВИ8.Видимость = Ложь;

		// а потом будем постепенно показывать только те, которые нужно (в
зависимости от максимальной приоритетности)
		Если МаксимальнаяПриоритетность > 0 Тогда
			Элементы.АбитуриентыБаллыЗаВИ1.Видимость = Истина;
		КонецЕсли;
		Если МаксимальнаяПриоритетность > 1 Тогда
			Элементы.АбитуриентыБаллыЗаВИ2.Видимость = Истина;
		КонецЕсли;
		Если МаксимальнаяПриоритетность > 2 Тогда
			Элементы.АбитуриентыБаллыЗаВИ3.Видимость = Истина;
		КонецЕсли;
		Если МаксимальнаяПриоритетность > 3 Тогда
			Элементы.АбитуриентыБаллыЗаВИ4.Видимость = Истина;
		КонецЕсли;
		Если МаксимальнаяПриоритетность > 4 Тогда
			Элементы.АбитуриентыБаллыЗаВИ5.Видимость = Истина;
		КонецЕсли;
		Если МаксимальнаяПриоритетность > 5 Тогда
			Элементы.АбитуриентыБаллыЗаВИ6.Видимость = Истина;
		КонецЕсли;
		Если МаксимальнаяПриоритетность > 6 Тогда
			Элементы.АбитуриентыБаллыЗаВИ7.Видимость = Истина;
		КонецЕсли;
		Если МаксимальнаяПриоритетность > 7 Тогда
			Элементы.АбитуриентыБаллыЗаВИ8.Видимость = Истина;
		КонецЕсли;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОбразовательнаяПрограммаПриИзменении(Элемент)
	// определяем доступность функции формирования списка абитуриентов
	ОбразовательнаяПрограмма = Объект.ОбразовательнаяПрограмма;
	Элемент = Элементы.Найти("СформироватьСписокАбитуриентов");
	Если ЗначениеЗаполнено(ОбразовательнаяПрограмма) Тогда
```

Продолжение листинга 17

```
			Элемент.Доступность = Истина;
		Иначе
			Элемент.Доступность = Ложь;
		КонецЕсли;
КонецПроцедуры

&НаСервереБезКонтекста
// функция, в которой будем получать максимальную приоритетность ВИ образовательной
программы
Функция ПолучитьМаксимальнуюПриоритетность(ОбразовательнаяПрограмма)
		// определим вступительные испытания образовательной программы
		Запрос = Новый Запрос;
		Запрос.Текст =
			"ВЫБРАТЬ
			|	ОбразовательныеПрограммы.ВступительныеИспытания.(
			|		ВступительноеИспытание КАК ВступительноеИспытание,
			|		Приоритетность КАК Приоритетность
			|	) КАК ВступительныеИспытания
			|ИЗ
			|	Справочник.ОбразовательныеПрограммы КАК ОбразовательныеПрограммы
			|ГДЕ
			|	ОбразовательныеПрограммы.Код = &Код";
		Запрос.УстановитьПараметр("Код", Строка(ОбразовательнаяПрограмма));
		РезультатЗапроса = Запрос.Выполнить();

		// пройдёмся по вступительным испытаниям из полученной выборки, чтобы определить
максимальную приоритетность ВИ ОП
		МаксимальнаяПриоритетностьВИ = 0;
		ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
		Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
			ВступительныеИспытания =
ВыборкаДетальныеЗаписи.ВступительныеИспытания.Выбрать();
			Пока ВступительныеИспытания.Следующий() Цикл
				Если ВступительныеИспытания.Приоритетность >
МаксимальнаяПриоритетностьВИ Тогда
					МаксимальнаяПриоритетностьВИ =
ВступительныеИспытания.Приоритетность;
				КонецЕсли;
			КонецЦикла;
		КонецЦикла;

		Возврат МаксимальнаяПриоритетностьВИ;
КонецФункции

&НаСервере
// функция получения количества мест по данной ОП
Функция ПолучитьКоличествоМест(ОбразовательнаяПрограмма)
		Возврат
Справочники.ОбразовательныеПрограммы.НайтиПоКоду(ОбразовательнаяПрограмма).КоличествоМе
ст;
КонецФункции

&НаСервере
// функция формирования списка абитуриентов на сервере по данной ОП
Функция СформироватьСписокАбитуриентовНаСервере(ОбразовательнаяПрограмма)
		// в запросе получим абитуриентов и их баллы по данной ОП, при условии, что у
абитуриента есть согласие на зачисление на данную ОП
		Запрос = Новый Запрос;
		Запрос.Текст =
			"ВЫБРАТЬ
```

Продолжение листинга 17

```
            |      Абитуриенты.Ссылка КАК Абитуриент,
            |      Абитуриенты.ПреимущественноеПраво КАК ПреимущественноеПраво,
            |      БаллыАбитуриентов.Баллы1Приоритета КАК Баллы1Приоритета,
            |      БаллыАбитуриентов.Баллы2Приоритета КАК Баллы2Приоритета,
            |      БаллыАбитуриентов.Баллы3Приоритета КАК Баллы3Приоритета,
            |      БаллыАбитуриентов.Баллы4Приоритета КАК Баллы4Приоритета,
            |      БаллыАбитуриентов.Баллы5Приоритета КАК Баллы5Приоритета,
            |      БаллыАбитуриентов.Баллы6Приоритета КАК Баллы6Приоритета,
            |      БаллыАбитуриентов.Баллы7Приоритета КАК Баллы7Приоритета,
            |      БаллыАбитуриентов.Баллы8Приоритета КАК Баллы8Приоритета,
            |      БаллыАбитуриентов.СуммаБалловЗаИД КАК СуммаБалловЗаИД,
            |      БаллыАбитуриентов.СуммаБаллов КАК СуммаБаллов
            |ИЗ
            |      РегистрСведений.БаллыАбитуриентов КАК БаллыАбитуриентов
            |            ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Абитуриенты КАК Абитуриенты
            |                  ЛЕВОЕ СОЕДИНЕНИЕ Документ.СогласиеОЗачислении КАК
СогласиеОЗачисленииАбитуриента
            |                  ПО Абитуриенты.СогласиеОЗачислении =
СогласиеОЗачисленииАбитуриента.Ссылка
            |            ПО БаллыАбитуриентов.Абитуриент = Абитуриенты.Ссылка
            |ГДЕ
            |      Абитуриенты.ОбразовательныеПрограммы.ОбразовательнаяПрограмма =
&ОбразовательнаяПрограмма
            |      И СогласиеОЗачисленииАбитуриента.ОбразовательнаяПрограмма =
&ОбразовательнаяПрограмма
            |      И БаллыАбитуриентов.ОбразовательнаяПрограмма =
&ОбразовательнаяПрограмма
            |УПОРЯДОЧИТЬ ПО
            |      СуммаБаллов УБЫВ,
            |      Баллы1Приоритета УБЫВ,
            |      Баллы2Приоритета УБЫВ,
            |      Баллы3Приоритета УБЫВ,
            |      Баллы4Приоритета УБЫВ,
            |      Баллы5Приоритета УБЫВ,
            |      Баллы6Приоритета УБЫВ,
            |      Баллы7Приоритета УБЫВ,
            |      Баллы8Приоритета УБЫВ,
            |      ПреимущественноеПраво УБЫВ";

      Запрос.УстановитьПараметр("ОбразовательнаяПрограмма", ОбразовательнаяПрограмма);
      РезультатЗапроса = Запрос.Выполнить();
      ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

      // создаём массив, в который будем добавлять в качестве элементов структуры с
данными абитуриентов
      СписокАбитуриентовКЗачислению = Новый Массив;

      Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
            // создаём структуру, в которую будем записывать данные абитуриента для
формирования приказа
            ДанныеАбитуриента = Новый Структура;

            ДанныеАбитуриента.Вставить("Абитуриент",
ВыборкаДетальныеЗаписи.Абитуриент);
            ДанныеАбитуриента.Вставить("Баллы1Приоритета",
ВыборкаДетальныеЗаписи.Баллы1Приоритета);
            ДанныеАбитуриента.Вставить("Баллы2Приоритета",
ВыборкаДетальныеЗаписи.Баллы2Приоритета);
            ДанныеАбитуриента.Вставить("Баллы3Приоритета",
ВыборкаДетальныеЗаписи.Баллы3Приоритета);
```

Продолжение листинга 17

```
            ДанныеАбитуриента.Вставить("Баллы4Приоритета",
ВыборкаДетальныеЗаписи.Баллы4Приоритета);
            ДанныеАбитуриента.Вставить("Баллы5Приоритета",
ВыборкаДетальныеЗаписи.Баллы5Приоритета);
            ДанныеАбитуриента.Вставить("Баллы6Приоритета",
ВыборкаДетальныеЗаписи.Баллы6Приоритета);
            ДанныеАбитуриента.Вставить("Баллы7Приоритета",
ВыборкаДетальныеЗаписи.Баллы7Приоритета);
            ДанныеАбитуриента.Вставить("Баллы8Приоритета",
ВыборкаДетальныеЗаписи.Баллы8Приоритета);
            ДанныеАбитуриента.Вставить("СуммаБалловЗаИД",
ВыборкаДетальныеЗаписи.СуммаБалловЗаИД);
            ДанныеАбитуриента.Вставить("СуммаБаллов",
ВыборкаДетальныеЗаписи.СуммаБаллов);

            // добавляем созданную структуру в общий массив
            СписокАбитуриентовКЗачислению.Добавить(ДанныеАбитуриента);
      КонецЦикла;

      Возврат СписокАбитуриентовКЗачислению;
КонецФункции

&НаКлиенте
Процедура СформироватьСписокАбитуриентов(Команда)
      ОбразовательнаяПрограмма = Объект.ОбразовательнаяПрограмма;

      // очистим табличную часть перед заполнением с помощью кнопки
      Объект.Абитуриенты.Очистить();

      // обратимся к серверу и получим максимальную приоритетность ВИ по ОП
      МаксимальнаяПриоритетность =
ПолучитьМаксимальнуюПриоритетность(ОбразовательнаяПрограмма);

      // сначала скроем все столбцы с ВИ
      Элементы.АбитуриентыБаллыЗаВИ1.Видимость = Ложь;
      Элементы.АбитуриентыБаллыЗаВИ2.Видимость = Ложь;
      Элементы.АбитуриентыБаллыЗаВИ3.Видимость = Ложь;
      Элементы.АбитуриентыБаллыЗаВИ4.Видимость = Ложь;
      Элементы.АбитуриентыБаллыЗаВИ5.Видимость = Ложь;
      Элементы.АбитуриентыБаллыЗаВИ6.Видимость = Ложь;
      Элементы.АбитуриентыБаллыЗаВИ7.Видимость = Ложь;
      Элементы.АбитуриентыБаллыЗаВИ8.Видимость = Ложь;

      // а потом будем постепенно показывать только те, которые нужно (в зависимости от
максимальной приоритетности)
      Если МаксимальнаяПриоритетность > 0 Тогда
            Элементы.АбитуриентыБаллыЗаВИ1.Видимость = Истина;
      КонецЕсли;
      Если МаксимальнаяПриоритетность > 1 Тогда
            Элементы.АбитуриентыБаллыЗаВИ2.Видимость = Истина;
      КонецЕсли;
      Если МаксимальнаяПриоритетность > 2 Тогда
            Элементы.АбитуриентыБаллыЗаВИ3.Видимость = Истина;
      КонецЕсли;
      Если МаксимальнаяПриоритетность > 3 Тогда
            Элементы.АбитуриентыБаллыЗаВИ4.Видимость = Истина;
      КонецЕсли;
      Если МаксимальнаяПриоритетность > 4 Тогда
            Элементы.АбитуриентыБаллыЗаВИ5.Видимость = Истина;
      КонецЕсли;
```

Продолжение листинга 17

```
	Если МаксимальнаяПриоритетность > 5 Тогда
		Элементы.АбитуриентыБаллыЗаВИ6.Видимость = Истина;
	КонецЕсли;
	Если МаксимальнаяПриоритетность > 6 Тогда
		Элементы.АбитуриентыБаллыЗаВИ7.Видимость = Истина;
	КонецЕсли;
	Если МаксимальнаяПриоритетность > 7 Тогда
		Элементы.АбитуриентыБаллыЗаВИ8.Видимость = Истина;
	КонецЕсли;

	// обратимся к серверу и получим количество мест по ОП
	КоличествоМест = ПолучитьКоличествоМест(ОбразовательнаяПрограмма);

	// вызываем функцию для получения списка абитуриентов к зачислению по данной ОП
(в виде массива)
	СписокАбитуриентовКЗачислению =
СформироватьСписокАбитуриентовНаСервере(ОбразовательнаяПрограмма);

	// заполнение табличной части данными из массива с итоговым списком абитуриентов
	Для н = 0 по СписокАбитуриентовКЗачислению.ВГраница() Цикл
		Если (н + 1) <= КоличествоМест Тогда
			Строка = Объект.Абитуриенты.Добавить();
			Строка.Абитуриент = СписокАбитуриентовКЗачислению[н].Абитуриент;

			// реквизиты баллов за ВИ будем заполнять только в пределах
максимальной приоритетности
			Если МаксимальнаяПриоритетность > 0 Тогда
				Строка.БаллыЗаВИ1 =
СписокАбитуриентовКЗачислению[н].Баллы1Приоритета;
			КонецЕсли;
			Если МаксимальнаяПриоритетность > 1 Тогда
				Строка.БаллыЗаВИ2 =
СписокАбитуриентовКЗачислению[н].Баллы2Приоритета;
			КонецЕсли;
			Если МаксимальнаяПриоритетность > 2 Тогда
				Строка.БаллыЗаВИ3 =
СписокАбитуриентовКЗачислению[н].Баллы3Приоритета;
			КонецЕсли;
			Если МаксимальнаяПриоритетность > 3 Тогда
				Строка.БаллыЗаВИ4 =
СписокАбитуриентовКЗачислению[н].Баллы4Приоритета;
			КонецЕсли;
			Если МаксимальнаяПриоритетность > 4 Тогда
				Строка.БаллыЗаВИ5 =
СписокАбитуриентовКЗачислению[н].Баллы5Приоритета;
			КонецЕсли;
			Если МаксимальнаяПриоритетность > 5 Тогда
				Строка.БаллыЗаВИ6 =
СписокАбитуриентовКЗачислению[н].Баллы6Приоритета;
			КонецЕсли;
			Если МаксимальнаяПриоритетность > 6 Тогда
				Строка.БаллыЗаВИ7 =
СписокАбитуриентовКЗачислению[н].Баллы7Приоритета;
			КонецЕсли;
			Если МаксимальнаяПриоритетность > 7 Тогда
				Строка.БаллыЗаВИ8 =
СписокАбитуриентовКЗачислению[н].Баллы8Приоритета;
			КонецЕсли;

			Строка.БаллыЗаИД = СписокАбитуриентовКЗачислению[н].СуммаБалловЗаИД;
```

Окончание листинга 17

```
                Строка.СуммаБаллов = СписокАбитуриентовКЗачислению[н].СуммаБаллов;
            Иначе
                Прервать;
            КонецЕсли;
        КонецЦикла;
КонецПроцедуры

&НаКлиенте
// чтобы нельзя было указать дату, которая ещё не наступила
Процедура ДатаПриИзменении(Элемент)
        Если Объект.Дата > ТекущаяДата() Тогда
            Предупреждение("Нельзя указать ещё не наступившую дату!");
            Объект.Дата = Неопределено;
        КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ВыбратьФайл(Элемент, ВыбранноеЗначение, СтандартнаяОбработка) // файл - скан-
копия приказа о зачислении
        СтандартнаяОбработка = Ложь;
        Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
        Диалог.Заголовок = "Выберите файл";
        Диалог.Фильтр = "Документ PDF (*.pdf)|*.pdf";
        Диалог.МножественныйВыбор = Ложь;

        // если файл уже добавлен, получаем его путь и открываем Проводник по нему
        Если Не Объект.СканКопия = Неопределено Тогда
            ФайлPDF = Новый Файл(Объект.СканКопия);
            Если ФайлPDF.Существует() Тогда
                Диалог.ПолноеИмяФайла = ФайлPDF.ПолноеИмя;
                Диалог.Каталог = ФайлPDF.Путь;
            КонецЕсли;
        Иначе // открываем Проводник в корне диска C
            Диалог.ПолноеИмяФайла = "";
            Диалог.Каталог = "C:\";
        КонецЕсли;

        // если файл был выбран
        Если Диалог.Выбрать() Тогда
            Объект.СканКопия = Диалог.ПолноеИмяФайла; // записываем в реквизит полный
путь к файлу
            Элемент = Элементы.Найти("ОткрытьФайл");
            Элемент.Доступность = Истина; // после выбора файла можно сделать
доступным его просмотр
        КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОткрытьФайл(Команда) // файл - скан-копия приказа о зачислении
        ФайлPDF = Новый Файл(Объект.СканКопия);
        Если ФайлPDF.Существует() Тогда
            ЗапуститьПриложение(ФайлPDF.ПолноеИмя);
        Иначе
            Сообщение = Новый СообщениеПользователю;
            Сообщение.Текст = "Файл не существует!";
            Сообщение.Поле = "СканКопия";
            Сообщение.ПутьКДанным = "Объект";
            Сообщение.Сообщить();
        КонецЕсли;
КонецПроцедуры
```

Листинг 18 – Программный код запроса для формирования отчёта «Количество заявлений»

```
ВЫБРАТЬ
	ОбщиеОбразовательныеПрограммы.Код КАК Код,
	ОбщиеОбразовательныеПрограммы.Наименование КАК Наименование,
	КОЛИЧЕСТВО(Абитуриенты.Наименование) КАК КоличествоЗаявлений
ИЗ
	Справочник.ОбразовательныеПрограммы КАК ОбщиеОбразовательныеПрограммы
	ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Абитуриенты КАК Абитуриенты
	ПО ОбщиеОбразовательныеПрограммы.Ссылка =
Абитуриенты.ОбразовательныеПрограммы.ОбразовательнаяПрограмма
СГРУППИРОВАТЬ ПО
	ОбщиеОбразовательныеПрограммы.Код,
	ОбщиеОбразовательныеПрограммы.Наименование
```

Листинг 19 – Программный код запроса для формирования отчёта «Количество согласий о зачислении»

```
ВЫБРАТЬ
	ОбщиеОбразовательныеПрограммы.Код КАК Код,
	ОбщиеОбразовательныеПрограммы.Наименование КАК Наименование,
	КОЛИЧЕСТВО(СогласиеОЗачислении.Ссылка) КАК КоличествоСогласийОЗачислении
ИЗ
	Справочник.ОбразовательныеПрограммы КАК ОбщиеОбразовательныеПрограммы
	ЛЕВОЕ СОЕДИНЕНИЕ Документ.СогласиеОЗачислении КАК СогласиеОЗачислении
	ПО ОбщиеОбразовательныеПрограммы.Ссылка =
СогласиеОЗачислении.ОбразовательнаяПрограмма
СГРУППИРОВАТЬ ПО
	ОбщиеОбразовательныеПрограммы.Код,
	ОбщиеОбразовательныеПрограммы.Наименование
```

Листинг 20 – Программный код запросов для формирования отчёта «Конкурсные списки абитуриентов»

```
ВЫБРАТЬ
	ОбразовательныеПрограммы.Наименование КАК НаименованиеОбразовательнойПрограммы,
	ОбразовательныеПрограммы.УровеньОбразования КАК УровеньОбразования,
	ОбразовательныеПрограммы.ФормаОбучения КАК ФормаОбучения,
	ОбразовательныеПрограммы.ОснованиеПолученияОбразования КАК
ОснованиеПолученияОбразования,
	ОбразовательныеПрограммы.Квота КАК Квота,
	ОбразовательныеПрограммы.КоличествоМест КАК КоличествоМест
ИЗ
	Справочник.ОбразовательныеПрограммы КАК ОбразовательныеПрограммы
ГДЕ
	ОбразовательныеПрограммы.Ссылка = &ОбразовательнаяПрограмма

ВЫБРАТЬ
	Абитуриенты.Ссылка КАК Абитуриент,
	Абитуриенты.Наименование КАК ФИОАбитуриента,
	БаллыАбитуриентов.СуммаБаллов КАК СуммаБалловАбитуриента,
	БаллыАбитуриентов.Баллы1Приоритета КАК Баллы1Приоритета,
	БаллыАбитуриентов.Баллы2Приоритета КАК Баллы2Приоритета,
	БаллыАбитуриентов.Баллы3Приоритета КАК Баллы3Приоритета,
	БаллыАбитуриентов.Баллы4Приоритета КАК Баллы4Приоритета,
	БаллыАбитуриентов.Баллы5Приоритета КАК Баллы5Приоритета,
	БаллыАбитуриентов.Баллы6Приоритета КАК Баллы6Приоритета,
	БаллыАбитуриентов.Баллы7Приоритета КАК Баллы7Приоритета,
	БаллыАбитуриентов.Баллы8Приоритета КАК Баллы8Приоритета,
	БаллыАбитуриентов.СуммаБалловЗаИД КАК СуммаБалловЗаИД,
	Абитуриенты.ПреимущественноеПраво КАК ПреимущественноеПраво,
```

Продолжение листинга 20

```
	Абитуриенты.СогласиеОЗачислении КАК СогласиеОЗачислении
ИЗ
	РегистрСведений.БаллыАбитуриентов КАК БаллыАбитуриентов
		ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Абитуриенты КАК Абитуриенты
		ПО БаллыАбитуриентов.Абитуриент = Абитуриенты.Ссылка
ГДЕ
	Абитуриенты.ОбразовательныеПрограммы.ОбразовательнаяПрограмма =
&ОбразовательнаяПрограмма
	И БаллыАбитуриентов.ОбразовательнаяПрограмма = &ОбразовательнаяПрограмма
	И
		ВЫБОР
			КОГДА &ТолькоССогласием = ИСТИНА ТОГДА
				Абитуриенты.СогласиеОЗачислении.ОбразовательнаяПрограмма =
&ОбразовательнаяПрограмма
			ИНАЧЕ ИСТИНА
		КОНЕЦ
```

Листинг 21 – Программный код модуля формы отчёта отчёта «Конкурсные списки абитуриентов»

```
&НаСервере
// передаём параметры в СКД при изменении поля ОП
Процедура ОбразовательнаяПрограммаПриИзмененииНаСервере(ОбразовательнаяПрограмма,
ТолькоССогласием)
	ЭтаФорма.Отчет.КомпоновщикНастроек.Настройки.ПараметрыДанных.УстановитьЗначениеПа
раметра("ОбразовательнаяПрограмма", ОбразовательнаяПрограмма);
	ЭтаФорма.Отчет.КомпоновщикНастроек.Настройки.ПараметрыДанных.УстановитьЗначениеПа
раметра("ТолькоССогласием", ТолькоССогласием);
КонецПроцедуры

&НаКлиенте
// вызов описанной выше функции на сервере
Процедура ОбразовательнаяПрограммаПриИзменении(Элемент)
	ОбразовательнаяПрограмма = Отчет.ОбразовательнаяПрограмма;
	ТолькоССогласием = Отчет.ТолькоССогласием;
	ОбразовательнаяПрограммаПриИзмененииНаСервере(ОбразовательнаяПрограмма,
ТолькоССогласием);
КонецПроцедуры

&НаСервере
// передаём параметры в СКД при изменении поля "Только с согласием о зачислении"
Процедура ТолькоССогласиемПриИзмененииНаСервере(ОбразовательнаяПрограмма,
ТолькоССогласием)
	ЭтаФорма.Отчет.КомпоновщикНастроек.Настройки.ПараметрыДанных.УстановитьЗначениеПа
раметра("ОбразовательнаяПрограмма", ОбразовательнаяПрограмма);
	ЭтаФорма.Отчет.КомпоновщикНастроек.Настройки.ПараметрыДанных.УстановитьЗначениеПа
раметра("ТолькоССогласием", ТолькоССогласием);
КонецПроцедуры

&НаКлиенте
// вызов описанной выше функции на сервере
Процедура ТолькоССогласиемПриИзменении(Элемент)
	ОбразовательнаяПрограмма = Отчет.ОбразовательнаяПрограмма;
	ТолькоССогласием = Отчет.ТолькоССогласием;
	ОбразовательнаяПрограммаПриИзмененииНаСервере(ОбразовательнаяПрограмма,
ТолькоССогласием);
КонецПроцедуры
```

Листинг 22 – Программный код запроса для формирования отчёта «Половое распределение заявлений по каждой образовательной программе»

```
ВЫБРАТЬ
	ОбщиеОбразовательныеПрограммы.Код КАК Код,
	ОбщиеОбразовательныеПрограммы.Наименование КАК Наименование,
	Абитуриенты.Пол КАК Пол,
	КОЛИЧЕСТВО(Абитуриенты.Ссылка) КАК КоличествоЗаявлений
ИЗ
	Справочник.ОбразовательныеПрограммы КАК ОбщиеОбразовательныеПрограммы
	ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Абитуриенты КАК Абитуриенты
	ПО ОбщиеОбразовательныеПрограммы.Ссылка =
Абитуриенты.ОбразовательныеПрограммы.ОбразовательнаяПрограмма
СГРУППИРОВАТЬ ПО
	ОбщиеОбразовательныеПрограммы.Код,
	ОбщиеОбразовательныеПрограммы.Наименование,
	Абитуриенты.Пол
```

Листинг 23 – Программный код запроса для формирования отчёта «Распределение по уровням образования для поступления»

```
ВЫБРАТЬ
	УровниОбразования.Ссылка КАК УровеньОбразования,
	КОЛИЧЕСТВО(Абитуриенты.Наименование) КАК КоличествоАбитуриентов
ИЗ
	Перечисление.УровниОбразования КАК УровниОбразования
	ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Абитуриенты КАК Абитуриенты
	ПО УровниОбразования.Ссылка = Абитуриенты.УровеньОбразованияДляПоступления
ГДЕ
	НЕ УровниОбразования.Ссылка =
ЗНАЧЕНИЕ(Перечисление.УровниОбразования.Бакалавриат)
	И НЕ УровниОбразования.Ссылка =
ЗНАЧЕНИЕ(Перечисление.УровниОбразования.Специалитет)
СГРУППИРОВАТЬ ПО
	УровниОбразования.Ссылка
```

Листинг 24 – Программный код запроса для формирования отчёта «Текущий средний балл по каждой образовательной программе»

```
ВЫБРАТЬ
	ОбщиеОбразовательныеПрограммы.Код КАК Код,
	ОбщиеОбразовательныеПрограммы.Наименование КАК Наименование,
	ВЫРАЗИТЬ(СРЕДНЕЕ(БаллыАбитуриентов.СуммаБаллов) КАК (ЧИСЛО(3, 0))) КАК
СреднийБалл
ИЗ
	Справочник.ОбразовательныеПрограммы КАК ОбщиеОбразовательныеПрограммы
		ЛЕВОЕ СОЕДИНЕНИЕ Документ.СогласиеОЗачислении КАК ОбщиеСогласиеОЗачислении
		ПО (ОбщиеСогласиеОЗачислении.ОбразовательнаяПрограмма =
ОбщиеОбразовательныеПрограммы.Ссылка)
		ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Абитуриенты КАК Абитуриенты
		ПО (Абитуриенты.ОбразовательныеПрограммы.ОбразовательнаяПрограмма =
ОбщиеОбразовательныеПрограммы.Ссылка)
			И (ОбщиеСогласиеОЗачислении.Ссылка =
Абитуриенты.СогласиеОЗачислении)
		ЛЕВОЕ СОЕДИНЕНИЕ РегистрСведений.БаллыАбитуриентов КАК БаллыАбитуриентов
		ПО (БаллыАбитуриентов.Абитуриент = Абитуриенты.Ссылка)
			И (БаллыАбитуриентов.ОбразовательнаяПрограмма =
ОбщиеОбразовательныеПрограммы.Ссылка)
```

Продолжение листинга 24

```
СГРУППИРОВАТЬ ПО
	ОбщиеОбразовательныеПрограммы.Код,
	ОбщиеОбразовательныеПрограммы.Наименование
```

Листинг 25 – Программный код модуля формы записи регистра сведений «Баллы абитуриентов»

```
&НаСервереБезКонтекста
// функция, в которой будем получать максимальную приоритетность ВИ образовательной программы
Функция ПолучитьМаксимальнуюПриоритетность(ОбразовательнаяПрограмма)
	// определим вступительные испытания образовательной программы
	Запрос = Новый Запрос;
	Запрос.Текст =
		"ВЫБРАТЬ
		|	ОбразовательныеПрограммы.ВступительныеИспытания.(
		|		ВступительноеИспытание КАК ВступительноеИспытание,
		|		Приоритетность КАК Приоритетность
		|	) КАК ВступительныеИспытания
		|ИЗ
		|	Справочник.ОбразовательныеПрограммы КАК ОбразовательныеПрограммы
		|ГДЕ
		|	ОбразовательныеПрограммы.Код = &Код";
	Запрос.УстановитьПараметр("Код", Строка(ОбразовательнаяПрограмма));
	РезультатЗапроса = Запрос.Выполнить();

	// пройдёмся по вступительным испытаниям из полученной выборки, чтобы определить максимальную приоритетность ВИ ОП
	МаксимальнаяПриоритетностьВИ = 0;
	ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
	Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
		ВступительныеИспытания = ВыборкаДетальныеЗаписи.ВступительныеИспытания.Выбрать();
		Пока ВступительныеИспытания.Следующий() Цикл
			Если ВступительныеИспытания.Приоритетность > МаксимальнаяПриоритетностьВИ Тогда
				МаксимальнаяПриоритетностьВИ = ВступительныеИспытания.Приоритетность;
			КонецЕсли;
		КонецЦикла;
	КонецЦикла;

	Возврат МаксимальнаяПриоритетностьВИ;
КонецФункции

&НаКлиенте
Процедура ПриОткрытии(Отказ)
	// если при открытии формы записи (уже созданной) ОП заполнена
	Если ЗначениеЗаполнено(Запись.ОбразовательнаяПрограмма) Тогда
		// получаем максимальную приоритетность ВИ образовательной программы
		МаксимальнаяПриоритетность = ПолучитьМаксимальнуюПриоритетность(Запись.ОбразовательнаяПрограмма);

		// сначала скроем все поля для добавления баллов за ВИ
		Элементы.Баллы1Приоритета.Видимость = Ложь;
		Элементы.Баллы2Приоритета.Видимость = Ложь;
		Элементы.Баллы3Приоритета.Видимость = Ложь;
		Элементы.Баллы4Приоритета.Видимость = Ложь;
		Элементы.Баллы5Приоритета.Видимость = Ложь;
```

Продолжение листинга 25

```
            Элементы.Баллы6Приоритета.Видимость = Ложь;
            Элементы.Баллы7Приоритета.Видимость = Ложь;
            Элементы.Баллы8Приоритета.Видимость = Ложь;

            // а потом будем постепенно показывать только те, которые нужно
            Если МаксимальнаяПриоритетность > 0 Тогда
                  Элементы.Баллы1Приоритета.Видимость = Истина;
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 1 Тогда
                  Элементы.Баллы2Приоритета.Видимость = Истина;
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 2 Тогда
                  Элементы.Баллы3Приоритета.Видимость = Истина;
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 3 Тогда
                  Элементы.Баллы4Приоритета.Видимость = Истина;
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 4 Тогда
                  Элементы.Баллы5Приоритета.Видимость = Истина;
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 5 Тогда
                  Элементы.Баллы6Приоритета.Видимость = Истина;
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 6 Тогда
                  Элементы.Баллы7Приоритета.Видимость = Истина;
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 7 Тогда
                  Элементы.Баллы8Приоритета.Видимость = Истина;
            КонецЕсли;
      КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОбразовательнаяПрограммаПриИзменении(Элемент)
      // получаем максимальную приоритетность ВИ образовательной программы
      МаксимальнаяПриоритетность =
ПолучитьМаксимальнуюПриоритетность(Запись.ОбразовательнаяПрограмма);

      // сначала скроем все поля для добавления баллов за ВИ
      Элементы.Баллы1Приоритета.Видимость = Ложь;
      Элементы.Баллы2Приоритета.Видимость = Ложь;
      Элементы.Баллы3Приоритета.Видимость = Ложь;
      Элементы.Баллы4Приоритета.Видимость = Ложь;
      Элементы.Баллы5Приоритета.Видимость = Ложь;
      Элементы.Баллы6Приоритета.Видимость = Ложь;
      Элементы.Баллы7Приоритета.Видимость = Ложь;
      Элементы.Баллы8Приоритета.Видимость = Ложь;

      // а потом будем постепенно показывать только те, которые нужно
      Если МаксимальнаяПриоритетность > 0 Тогда
            Элементы.Баллы1Приоритета.Видимость = Истина;
      КонецЕсли;
      Если МаксимальнаяПриоритетность > 1 Тогда
            Элементы.Баллы2Приоритета.Видимость = Истина;
      КонецЕсли;
      Если МаксимальнаяПриоритетность > 2 Тогда
            Элементы.Баллы3Приоритета.Видимость = Истина;
      КонецЕсли;
      Если МаксимальнаяПриоритетность > 3 Тогда
            Элементы.Баллы4Приоритета.Видимость = Истина;
```

Окончание листинга 25

```
	КонецЕсли;
	Если МаксимальнаяПриоритетность > 4 Тогда
		Элементы.Баллы5Приоритета.Видимость = Истина;
	КонецЕсли;
	Если МаксимальнаяПриоритетность > 5 Тогда
		Элементы.Баллы6Приоритета.Видимость = Истина;
	КонецЕсли;
	Если МаксимальнаяПриоритетность > 6 Тогда
		Элементы.Баллы7Приоритета.Видимость = Истина;
	КонецЕсли;
	Если МаксимальнаяПриоритетность > 7 Тогда
		Элементы.Баллы8Приоритета.Видимость = Истина;
	КонецЕсли;
КонецПроцедуры

&НаСервере
// вызов процедуры проверки заполнения из модуля набора записей
Процедура ПередЗаписьюНаСервере(Отказ, ТекущийОбъект)
	Документ = РеквизитФормыВЗначение("Запись");
	Документ.ПроверитьЗаполнение();
КонецПроцедуры
```

Листинг 26 – Программный код модуля набора записей регистра сведений «Баллы абитуриентов»

```
Процедура ОбработкаПроверкиЗаполнения(Отказ, ПроверяемыеРеквизиты) Экспорт
	Для каждого Объект из ЭтотОбъект Цикл
		// определим вступительные испытания образовательной программы
		Запрос = Новый Запрос;
		Запрос.Текст =
			"ВЫБРАТЬ
			|	ОбразовательныеПрограммы.ВступительныеИспытания.(
			|		ВступительноеИспытание КАК ВступительноеИспытание,
			|		Приоритетность КАК Приоритетность
			|	) КАК ВступительныеИспытания
			|ИЗ
			|	Справочник.ОбразовательныеПрограммы КАК
ОбразовательныеПрограммы
			|ГДЕ
			|	ОбразовательныеПрограммы.Код = &Код";
		Запрос.УстановитьПараметр("Код", Строка(Объект.ОбразовательнаяПрограмма));
		РезультатЗапроса = Запрос.Выполнить();

		// пройдёмся по вступительным испытаниям из полученной выборки, чтобы
определить максимальную приоритетность ВИ ОП
		МаксимальнаяПриоритетность = 0;
		ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
		Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
			ВступительныеИспытания =
ВыборкаДетальныеЗаписи.ВступительныеИспытания.Выбрать();
			Пока ВступительныеИспытания.Следующий() Цикл
				Если ВступительныеИспытания.Приоритетность >
МаксимальнаяПриоритетность Тогда
					МаксимальнаяПриоритетность =
ВступительныеИспытания.Приоритетность;
				КонецЕсли;
			КонецЦикла;
		КонецЦикла;
```

Продолжение листинга 26

```
            // добавляем реквизиты в проверяемые в зависимости от их количества
            Если МаксимальнаяПриоритетность > 0 Тогда
                  ПроверяемыеРеквизиты.Добавить("Баллы1Приоритета");
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 1 Тогда
                  ПроверяемыеРеквизиты.Добавить("Баллы2Приоритета");
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 2 Тогда
                  ПроверяемыеРеквизиты.Добавить("Баллы3Приоритета");
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 3 Тогда
                  ПроверяемыеРеквизиты.Добавить("Баллы4Приоритета");
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 4 Тогда
                  ПроверяемыеРеквизиты.Добавить("Баллы5Приоритета");
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 5 Тогда
                  ПроверяемыеРеквизиты.Добавить("Баллы6Приоритета");
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 6 Тогда
                  ПроверяемыеРеквизиты.Добавить("Баллы7Приоритета");
            КонецЕсли;
            Если МаксимальнаяПриоритетность > 7 Тогда
                  ПроверяемыеРеквизиты.Добавить("Баллы8Приоритета");
            КонецЕсли;
      КонецЦикла;
КонецПроцедуры
```