КУРСОВАЯ РАБОТА

“Обработка биологических сигналов в Delphi”

Дисциплина: “Программирование и основы алгоритмизации”

Работу выполнил студент магистратуры:

Мадиярова Д.Т. 70М – 13 ТБАТМ (узб)

Проверил преподаватель:

Васильева С.А.

Ташкент - 2014

Содержание

Введение

В настоящее время одними из самых распространенных являются сердечно-сосудистые заболевания. Поэтому особенно важна ранняя диагностика заболеваний сердечно-сосудистой системы. Особое значение приобретает изучение центральной гемодинамики (кровообращения) при таких экстремальных ситуациях, как гипертонический криз, кардиогенный шок, инфаркт миокарда и т.д. При этом ведущая роль принадлежит гемодинамическим показателям, таким как ударный объем и общее периферическое сопротивление.

Современные медицинские приборы используют микропроцессорную обработку получаемых сигналов. Для этого требуется соответственное программное обеспечение. В данной курсовой представлено описание программы для анализа сигналов диагностики сердечно-сосудистой системы, созданной в среде программирования Delphi.

1 Основная часть

1.1 Задание

Разместить на форму компонент Диаологовое Окно Выбора Файла (OpenDialog), компонент Изображение (Image), четыре кнопки (Button), несколько текстовых полей (Edit), компонент Главное меню (MainMenu), компонент Контекстное меню (PopupMenu), компонент Список действий (ActionList).

По щелчку на кнопке *Открыть файл* выбрать из диалогового окна, созданного с помощью компонента OpenDialog, один из файлов. Использовать обработку исключительных ситуаций для открытия файла.

По щелчку на кнопке *Данные* вывести в текстовые поля Edit данные о пациенте.

По щелчку на кнопке *Разместить* разместить R-зубцы на ЭКГ, точки максимума дифференцированной реограммы и точки начала изгнания крови из левого желудочка сердца.

По щелчку на кнопке *Печать* вывести на печать содержимое окна программы (работа с канвой принтера).

Те же действия осуществлять при выборе соответствующих пунктов главного и контекстного меню (использовать компонент Список действий ActionList).

Написать теоретическую часть: реографическое исследование, алгоритмы определения характерных точек на ЭКГ и реограмме.

Создать формальное описание программы.

1.2 Обзор литературы

Определение гемодинамических показателей проводится инвазивными и неинвазивными методами.

Инвазивный метод связан с нарушением кожного покрова. Он связан с определенным риском для пациента и не всегда приемлем.

При неинвазивных способах нарушения кожного покрова не происходит. Эти способы просты для медицинского персонала и необременительны для пациента, поэтому, по возможности, предпочтительней применять неинвазивные методы исследования.

Одним из неинвазивных методов является реография (импедансная плетизмография). Реография - практически единственный метод, обеспечивающий экспресс-диагностику состояния центральной гемодинамики в условиях атравматичного и необременительного для больного исследования.

Реография — неинвазивный метод исследования кровоснабжения органов, в основе которого лежит принцип регистрации изменений электрического сопротивления тканей в связи с меняющимся кровенаполнением. Чем больше приток крови к тканям, тем меньше их сопротивление. Для получения реограммы через тело пациента пропускают переменный ток частотой 50-100кГц, малой силы (не более 10 мкА), создаваемый специальным генератором.

Принципиальная разработка реографической методики принадлежит Н. Манну (1937). В дальнейшем методика (электроплетизмография, импеданс-плетизмография) получила развитие в работах А. А. Кедрова и Т. Ю. Либермана (1941— 1949) и др. Детальная разработка и внедрение в клиническую практику метода реографии связано с именами австрийских исследователей W. Holzer, К. Polzer и А. Marko. Им же принадлежит по существу первая монография (Rheokardiographie, Wien, 1946), в которой авторы не только осветили технические стороны метода (электрические схемы аппарата, варианты генератора переменного тока и другие), но и представили результаты клинического использования реографии при различных заболеваниях сердечно сосудистой системы. Существенный вклад в разработку метода реографии внес Ю.Т. Пушкарь, создавший отечественную конструкцию аппарата и изменивший методику регистрации реограммы

(прекардильная реокардиография). В настоящее время доказано клиническое значение применения метода реографии.

В зависимости от конкретной клинической задачи меняется зона исследования, и соответственно место наложения электродов. Поэтому различают реографию легких, сосудов мозга (реоэнцефалография), сосудов конечностей (реовазография) и другие.

Принципиальной основой метода реографии является зависимость изменений сопротивления от изменений кровенаполнения в изучаемом участке тела человека. Другими словами, изучаются пульсовые колебания электрического сопротивления.

Более полное представление о пульсовых колебаниях электрического сопротивления получают при учете (соотношении) базового сопротивления исследуемого участка (то есть суммарного сопротивления тела зондирующему току с частотой 50—100 кГц). Полный импеданс (сопротивление) состоит из двух величин, постоянный или базовый импеданс, обусловленный общим кровенаполнением тканей и их сопротивлением, и переменный или пульсовой импеданс, вызванный колебаниями кровенаполнения во время сердечного цикла. Величина пульсового импеданса ничтожно мала и составляет не более 0,5 % общего импеданса. Вместе с тем пульсовой импеданс составляет объект изучения для реографии.

Регистрация реограмм осуществляется с помощью реографов. Последние состоят из следующих элементов генератора высокой частоты, преобразователя «импеданс-напряжение», детектора, усилителя, калибровочного устройства, дифференцирующей цепочки.

При биполярной методике накладывают 2 электрода, каждый из которых одновременно является токовым и измерительным, электроды фиксируют на соответствующем участке тела. Для снижения контактного сопротивления между электродом и кожей используются те же приемы, что и при записи ЭКГ. При использовании тетраполярной методики участок исследования ограничивают парой измерительных электродов, а возникшее в них напряжение снимают с помощью другой пары электродов, расположенных кнаружи по отношению к первой (токовые). Тетраполярная методика более точна, ибо резко (до минимума) снижается влияние контактного сопротивления (нет необходимости накладывать прокладки, смоченные растворами солей или щелочей, а также пользоваться электродной пастой) и электродной поляризации. Это позволяет с высокой степенью точности измерить импеданс глубинных тканей. Кроме того, достаточно точно получаемые сведения о базисном импедансе позволяют дать количественную оценку основным гемодинамическим показателям ударному и минутному объемам кровообращения.

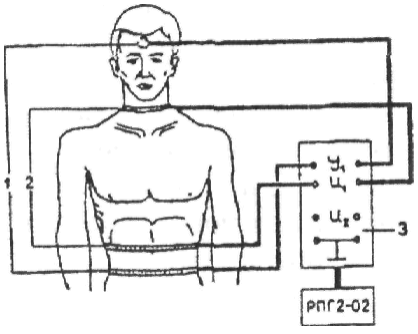

Рис. 1. Расположение электродов при проведении реографического исследования

Запись реограмм производится в теплом помещении через 1,5—2 ч после приема пищи или натощак, в положении лежа на спине после 15—20-минутного отдыха. Одновременно с двумя реограммами (основной и дифференциальной) записывается ЭКГ во II стандартном отведении и иногда ФКГ в V точке или над верхушкой на одном из среднечастотных диапазонов. Желательно регистрацию реограммы производить на задержке дыхания при неполном выдохе. Запись производят при скорости движения

лентопротяжного механизма 25—50 мм/с (реже — 100 мм/с). Необходимо следить за калибровочным сигналом (0,1 Ом=10 мм).

Количественный анализ предусматривает определение следующих показателей (рис. 2):

1. Амплитуда систолической волны в миллиметрах измеряется от основания систолической волны до высшей точки реограммы.
2. Амплитуда диастолической волны в миллиметрах измеряется от основания диастолической волны до высшей ее точки.
3. Реографический индекс (систолический - РСИ и диастолический - РДИ) - отношение систолической (диастолической) волны к стандартному калибровочному сигналу (0,1 Ом =10 мм), выражается в относительных единицах. Этот показатель характеризует величину и скорость притока (оттока) крови в исследуемой зоне. Амплитуда кривой измеряется от изолинии до высшей точки волны.

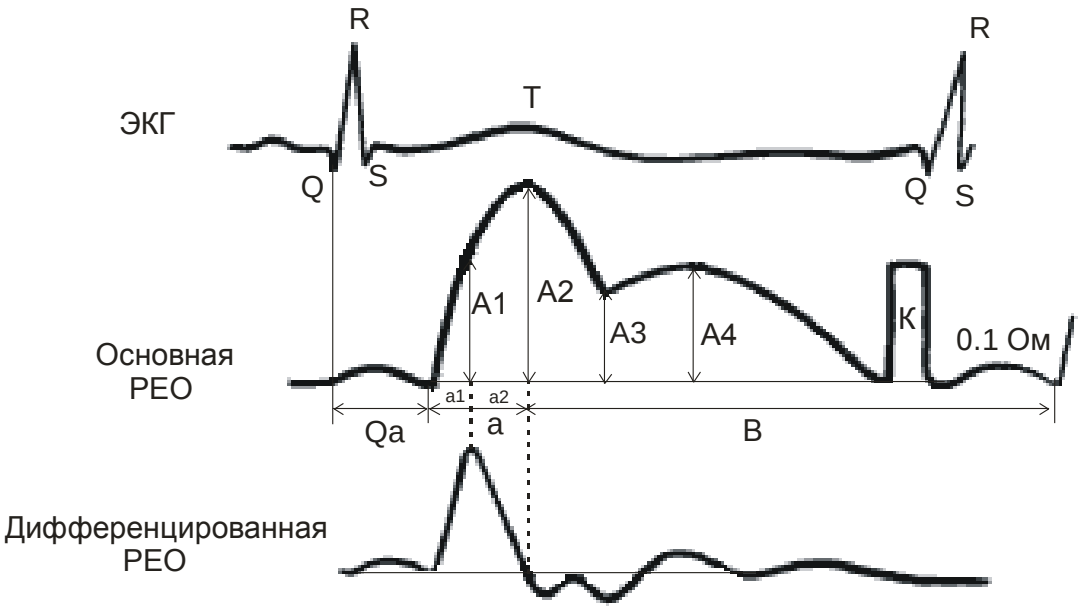

Рис. 2. Реографическая кривая с обозначением основных точек:
A1 – максимальная крутизна восходящего колена волны;
A2 – максимальная амплитуда систолической волны;
A3 – нижняя точка инцизуры;
A4 – максимальная амплитуда диастолической волны реограммы;
K – калибровочный сигнал;
B – длительность нисходящей части реограммы;
Qa – время распространения реографической волны на участке "сердце – исследуемый орган";
a – время максимального систолического наполнения сосудов;
a1 – время быстрого наполнения;
a2 – время медленного наполнения.

4. Интервал Qa или время распространения пульсовой волны (ВРПВ) на участке «сердце — исследуемый орган» в секундах — соответствует периоду напряжения при фазовом анализе систолы желудочков. Измеряется от начала зубца Q ЭКГ до начала волны реограммы, связанной с данным сердечным циклом. Интервал Qa уменьшается при повышении тонуса или склерозе магистральных сосудов

5. Период или время быстрого наполнения ($ВН_{быстр}$) - от начала подъема систолической волны реограммы до точки максимальной крутизны на ее восходящем колене (соответствует проекции вершины основного зубца дифференциальной реограммы на восходящее колено объемной реограммы). Этот показатель отражает величину ударного объема и функциональное состояние крупных сосудов.
6. Период или время медленного наполнения ($ВН_{медл}$) - от точки максимальной крутизны на восходящем колене реограммы до ее вершины. Этот показатель определяется также как разность между $ВН_{макс}$ и $ВН_{быстр}$ и отражает функциональное со стояние сосудов среднего и мелкого калибра.
7. $ВН_{быстр}$ и $ВН_{медл}$ составляют период максимального наполнения - $ВН_{макс}$ который измеряется от начала восходящей части кривой до ее вершины.
8. Амплитудно-частотный показатель (АЧП) — отношение реографического индекса (РИ) к длительности сердечного цикла R-R. АЧП характеризует величину объемного кровотока в исследуемой области в единицу времени.
9. Отношение амплитуд систолической и диастолической волн ($А_с/А_д$) отражает степень преобладания артериального притока во время систолы над венозным оттоком во время диастолы.
10. Время общего наполнения ($ВН_{общ}$) - интервал от начала подъема реограммы отражает общее время систолического притока крови в данную сосудистую область
11. Продолжительность катакроты в секундах (от высшей точки кривой реограммы до точки пересечения с изолинией)
12. Отношение времени восходящей части к времени нисходящей в процентах.
13. Отношение времени восходящей части реограммы к длительности сердечного цикла или к сумме, как показатель эластичности и тонуса сосудов.
14. Коэффициенты, отражающие отношение времени быстрого наполнения и времени медленного наполнения к общей длительности наполнения ($ВН_{быстр}$)/($ВН_{общ}$), ($ВН_{медл}$)/($ВН_{общ}$).

Следует заметить, что в реографии, как ни в одном из методов инструментальной диагностики сердечно-сосудистой системы нет единой методики количественных расчетов и нет единой терминологии. В каждом конкретном случае врач должен определить объем анализируемых показателей, который позволил бы при минимальных расчетах получить оптимальную информацию. [2]

1.3 Формальное описание программы

Данные сигналов и информация о пациенте располагается в типизированных *.dat файлах (тип запись).

Поля записи
RegNumber: smallint - регистрационный номер;
Da_ta, Name: shortstring - дата записи, ФИО;
Distance, Perimetr: smallint – расстояние между электродами, обхват грудной клетки;
Leng_s, Weight: smallint – рост, вес;
Sist_BP, Diast_BP: smallint – систолическое давление, диастолическое давление;
B: array[1..4,1..Predel] of smallint – массивы сигналов ЭКГ, реограммы, ФКГ, опорный сигнал;
Kalibr_Z0, Kalibr_DZ: Real48 – значения калибровки;
Константа
Predel=7000 (количество точек в массиве).

Процедуры и функции:

procedure Action1Execute(Sender: TObject); - процедура при нажатии на кнопке Открыть (п.1.4)
procedure Action2Execute(Sender: TObject); - процедура при нажатии на кнопке Данные - ввод информации в поля TEdit и построение графика (п.1.8)
procedure Action3Execute(Sender: TObject); - процедура при нажатии на кнопке Разместить (п.1.9, 1.10)
procedure Action4Execute(Sender: TObject); - процедура при нажатии на кнопке Печать (п.1.11)
procedure MaxR; - процедура поиска R-зубцов (п.1.9)
procedure MaxM; - процедура поиска максимумов дифференцированной реограммы (п.1.9)
procedure GelZ; - процедура поиска точек выброса крови из левого желудочка сердца (п.1.10)
procedure PrintGrafik; - печать графиков (ФКГ, Рео, ЭКГ) (п.1.11)
procedure PrintZ; - обозначение точек выброса крови из левого желудочка сердца при печати (п.1.10, 1.11)
procedure PrintR; - обозначение R-зубцов при печати (п.1.9, 1.11)
procedure PrintM; - обозначение максимумов дифференцированной реограммы при печати (п.1.9, 1.11)
function WidthX: Integer; - получение ширины страницы в пикселях для перевода координат при печати (п.1.7, 1.11)
function HeightY: Integer; - получение высоты страницы в пикселях для перевода координат при печати (п.1.7, 1.11)
function II(x: smallint):integer; - функция пересчёта координат по оси X (п.1.7)
function JJ(y,y1,y2: smallint):integer; - функция пересчёта координат по оси Y (п.1.7)
function MinEKG: smallint; - нахождение минимального значения ЭКГ (п.1.6)
function MaxEKG: smallint; - нахождение максимального значения ЭКГ (п.1.6)
function MinReo: smallint; - нахождение минимального значения дифференцированной реограммы (п.1.6)
function MaxReo: smallint; - нахождение максимального значения дифференцированной реограммы (п.1.6)
function MinFKG: smallint; - нахождение минимального значения ФКГ (п.1.6)
function MaxFKG: smallint; - нахождение максимального значения ФКГ (п.1.6)
function PorogR: smallint; - нахождение порогового значения (для поиска R-зубцов) (п.1.9)
function PorogM: smallint; - нахождение порогового значения (для поиска максимумов дифференцированной реограммы) (п.1.9)

Тип:

TReo=**record**
RegNumber: smallint - регистрационный номер;
Da_ta, Name: shortstring - дата записи, ФИО;
Distance, Perimetr: smallint – расстояние между электродами, обхват грудной клетки;
Leng_s, Weight: smallint – рост, вес;
Sist_BP, Diast_BP: smallint – систолическое давление, диастолическое давление;
B: array[1..4,1..Predel] of smallint – массивы сигналов ЭКГ, реограммы, ФКГ, опорный сигнал;
Kalibr_Z0, Kalibr_DZ: Real48 – значения калибровки;

Глобальные переменные:

Reo1: TReo; – переменная, в которую заносятся данные из файла
F: file of TReo; – файловая переменная (п.1.4.)
I1, I2: integer – размеры изображения по горизонтальной оси. Эти параметры необходимы для пересчёта координат. |I2-I1| - ширина изображения. (п.1.6.)
J1, J2: integer – размеры изображения по вертикальной оси. Эти параметры необходимы для пересчёта координат. |J2-J1| - высота изображения. (п.1.6.)
ARR: array[1..80] of smallint; - массив, в который вносятся абсциссы максимумов дифференцированной реограммы (п.1.10.)
h1: smallint; - сумма модулей глобального максимума и минимума графиков ЭКГ, РЕО и ФКГ
h2: smallint; - сумма модулей глобальных минимумов графиков ЭКГ, РЕО и глобального максимума ЭКГ
h2: smallint; - сумма модулей глобальных минимумов графиков ЭКГ, РЕО, ФКГ и глобальных максимумов РЕО и ЭКГ

} Необходимы для правильного расположения графиков по вертикали (п.1.7.)

Константа:

Predel=7000 - количество точек в массиве.

1.4 Работа с файлами

В типизированных файлах данные хранятся в виде структуры, представляющие какой-либо тип. В данном случае - запись. Для начала работы с файлом необходимо инициализировать файловую переменную этого типа (в данной программе она обозначается как F). Файловая переменная задаётся с помощью структуры file of.

F:=file of TReo.

Связывание файловой переменной с именем файла происходит при помощи процедуры AssignFile(<файловая переменная>,<название файла>). Файл открывается с помощью процедуры Reset(<файловая переменная>).

Считывание данных из файла переменной в переменную для сохранения прочитанных данных происходит с помощью переменной Read(<файловая переменная>,<переменная для сохранения непрочитанных данных>).

После считывания данных из файла в независимую переменную файл нужно закрыть с помощью метода CloseFile(<файловая переменная>).

1.5 Обработка исключительных ситуаций при открытии файла

Исключительные ситуации в среде программирования Delphi управляются с помощью нескольких операторов. В данной программе использовался оператор try…except.

Так как при открытии файла могут возникать различные исключительные ситуации, то был применён базовый класс исключений Exception.

```
try
  AssignFile(F,OpenDialog1.FileName);          ////Открытие файла
  Reset(F);
  Read(F,Reo1);                          ////Чтение данных из файла в переменную
  CloseFile(F);                          ////Закрытие файла
```

```
except
  on E:Exception do ShowMessage(E.Message);   /// вызов сообщения об ошибке
    end;
end;
```

Сообщение об ошибке изображено на рис.3

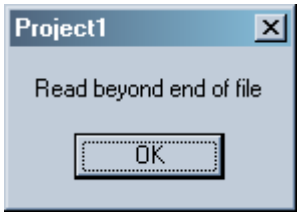

Рис.3. Уведомление об ошибке

При нажатии на кнопку ОК, окошко ошибки закрывается и основная программа снова становится доступна.

1.6 Функции нахождения глобального максимума и минимума сигналов

Глобальные максимумы понадобятся для пересчёта координат графиков в файле в координаты компонента Изображение (TImage).

Алгоритм работает по принципу последовательного сравнивания каждого значения из массива с некоторым максимальным (минимальным) значением. Если текущее значение больше(меньше) максимального(минимального), то это значение становится максимальным(минимальным).

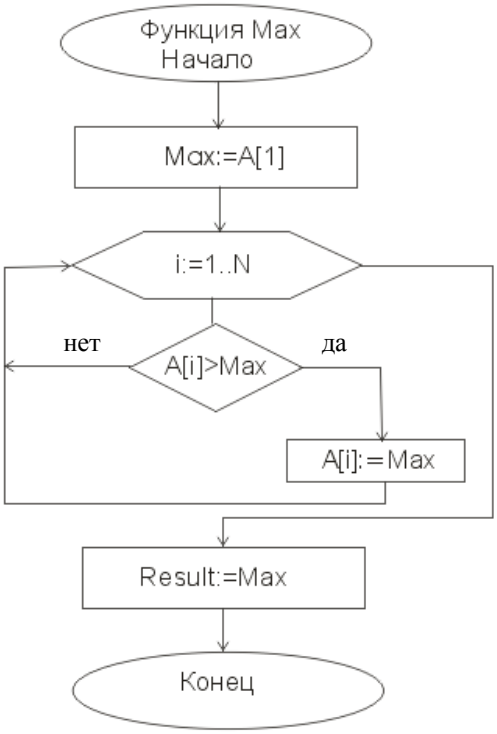

Рис.4 Блок-схема (для нахождения максимума):

A[i] – массив, в котором находится максимальное значение.

1.7 Функция пересчёта координат.

II и JJ – функция пересчёта координат (абсциссу и ординату, соответственно) к координатам на экране. Формула для преобразования координат получается с помощью свойства подобия. Необходимо также учитывать, что на экране ось ординат направлена вниз. На нижерасположенных рисунках показано, как необходимо делать преобразование координат для построения графиков зоне изображения Image1.

I1:=0; I2:=Image1.Width (ширина) – размеры изображения по горизонтальной оси

J1:=0; J2:=Image1.Height (высота) – размеры изображения по вертикальной оси.

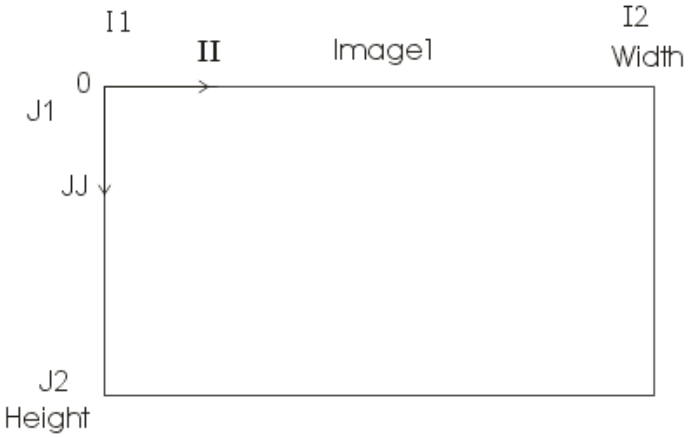

Рис.5. Координатные оси компонента TImage

x1:=0; x2:=Predel – количество элементов массива

y1:=0; y2:=|Max(Reo)|+|Min(Reo)|+|Max(FKG)|+|Min(FKG)|+|Max(EKG)|+|Min(EKG)| - сумма модулей глобальных максимумов и минимумов трёх графиков.

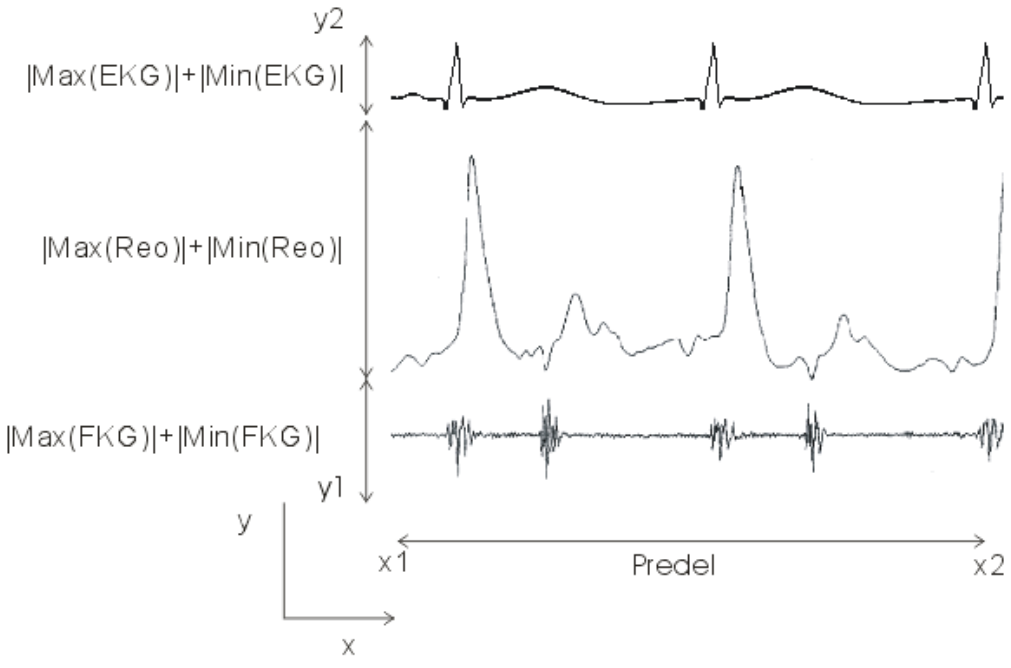

Рис.6. Координаты графиков в массиве

Из формулы подобия $\frac{x-x_1}{x_2-x_1}=const$ получаются следующие преобразования координат:

$$\frac{II-I_1}{I_2-I_1}=\frac{x-x_1}{x_2-x_1};\quad II=I_1+\frac{(I_2-I_1)(x-x_1)}{x_2-x_1}$$

$$\frac{JJ-J_1}{-(J_2-J_1)}=\frac{y-y_1}{y_2-y_1};\quad JJ=J_1+\frac{(J_1-J_2)(y-y_1)}{y_2-y_1}$$

1.8 Построение графиков по данным из массива

Если функция для построения графика задана в массиве (т.е. поточечно), то необходимо соединять последовательно пары точек i и i+1. Каждая точка имеет координаты: по оси абсцисс – порядковый номер точки в массиве, а по оси ординат – значение в массиве этой точки.

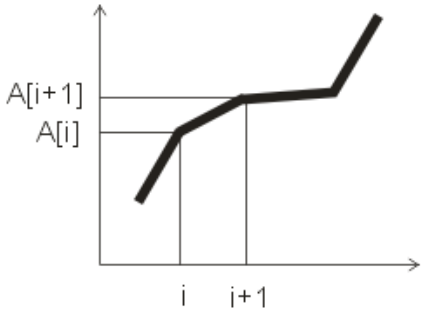

Рис.7. Принцип построения графика по точкам

В программе это можно сделать с помощью методов MoveTo(X,Y:integer) и LineTo(X,Y:integer). Метод MoveTo(X,Y:integer) устанавливает курсор на канву компонента TImage в точку с координатами X,Y. Метод LineTo(X,Y:integer) проводит линию от исходной точки к точке с координатами, указанными в LineTo.

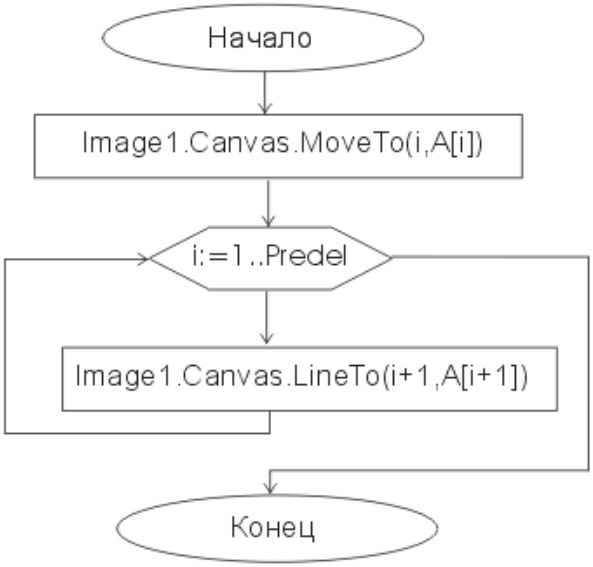

Рис.8. Блок-схема для алгоритма построения графика.

1.9 Нахождение R-зубцов на ЭКГ

Определить R-зубцы на ЭКГ или соответствующие им зубцы на дифференцированной ЭКГ можно с помощью следующего алгоритма. Вычисляется некий порог, и среди значений, которые его превышают, ищется локальный максимум.

Порог может вычисляться, например, как сумма среднего значения ЭКГ и половины максимального значения ЭКГ. Далее ищется первое значение, которое больше порога. От этого значения до первого значения меньшего, чем порог, ищется максимальное значение, которое и определяется как R-зубец (рис. 9, 10, 11). [1]

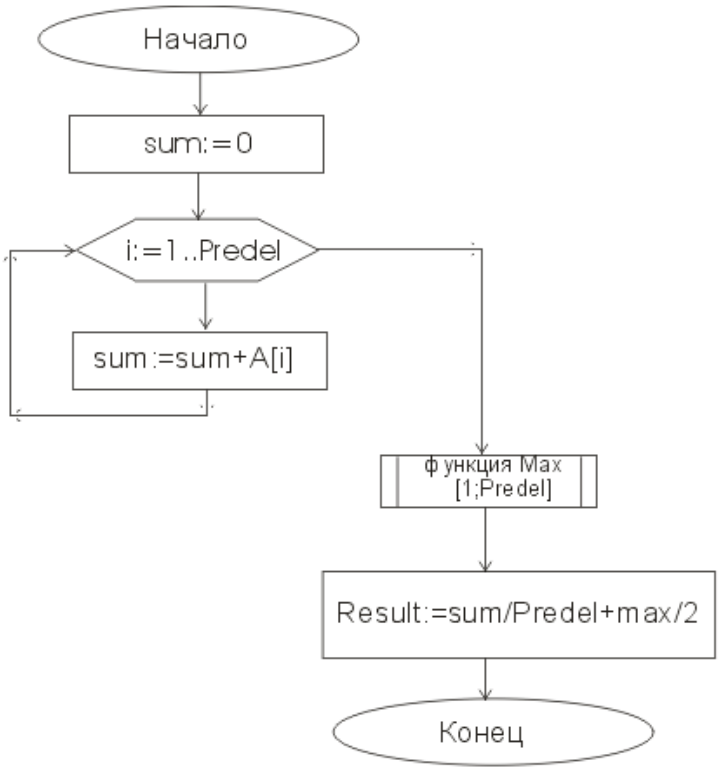

Рис. 9. Блок-схема для алгоритма нахождения порога

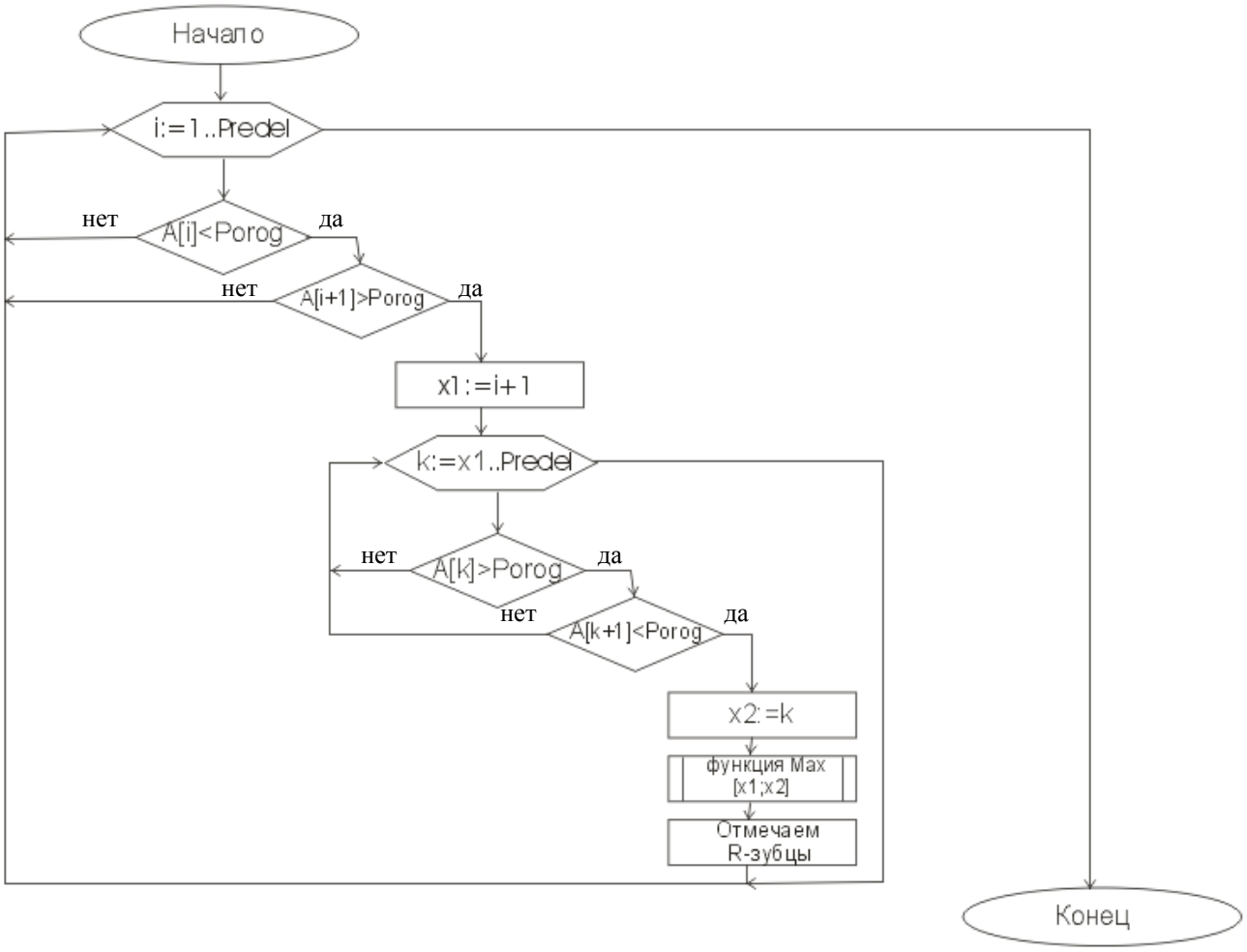

Рис. 10. Блок-схема для алгоритма нахождения R-зубцов

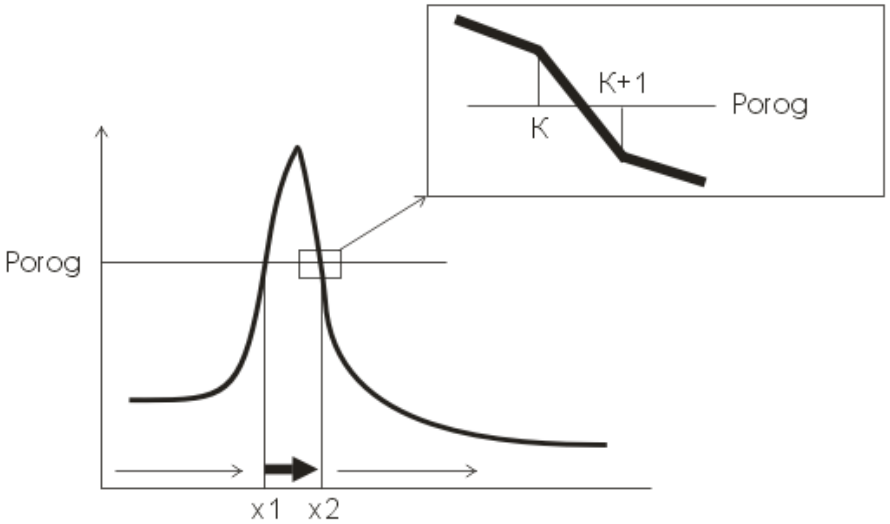

Рис.11 . Иллюстрация к алгоритму нахождения R-зубца.

Нахождение максимумов дифференцированной реограммы происходит аналогичным образом.

1.10 Определение точек начала изгнания крови из левого желудочка сердца

К кривой дифференцированной реограммы в точке максимума второй производной реограммы проводится касательная до пересечения ее с нулевым уровнем. Эта точка является абсциссой точки начала изгнания крови из левого желудочка сердца.

Для нахождения этих точек на графике необходимо рассматривать каждый период в отдельности. Поэтому предварительно необходимо разбить график на периодичные части. Точками разделения этих периодов могут служить максимумы дифференцированной реограммы. Их координаты абсцисс заносятся в статический массив на стадии нахождения максимумов дифференцированной реограммы. После нахождения координат происходит анализ реограммы на каждом отдельном периоде (рис. 12).

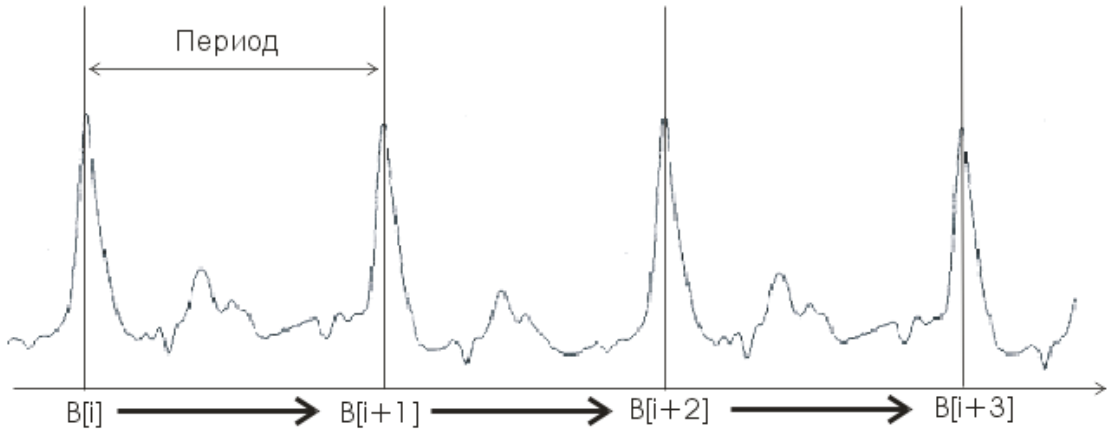

Рис. 12. Периодичность реограммы

Так как график функции строился по точкам и представляет собой ломанную, то провести касательную к одной точке не представляется возможным. Однако, по определению касательной можно воспользоваться двумя соседними точками и провести через них секущую (касательная является предельным случаем секущей, см. рис.13).

Геометрический смысл производной - коэффициент наклона прямой, т.е. тангенс угла наклона. Тангенс – это отношение противолежащего катета прямоугольного треугольника к прилежащему. Следовательно,

$$Y' = k = tg\ \alpha = \frac{A[i+1] - A[i-1]}{(i+1) - (i-1)} = \frac{A[i+1] - A[i-1]}{2}$$

Далее ищется максимальная производная на всем периоде с помощью функции Max (см. пункт 1.5). Абсцисса точки начала выброса определяется через тангенс угла. Чтобы компенсировать сдвиг между секущей и касательной (они параллельны), в формулу будет подставляться координата точки i.

Пусть t – абсцисса точки начала выброса крови. Она равна разности абсциссы точки максимума производной i и основания касательной к точке i.

$t = i - \frac{A[i]}{tg\ \alpha}$, где i – точка максимума второй производной основной реограммы.

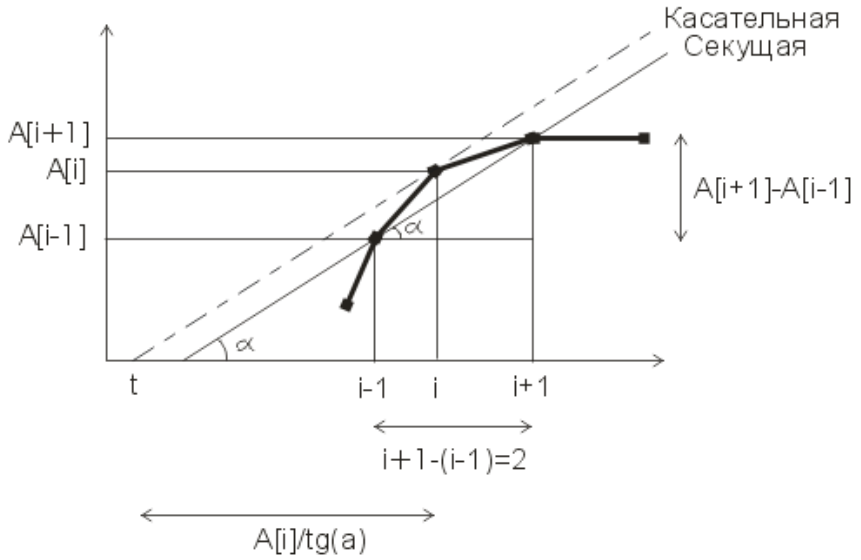

Рис. 13. Нахождение производной и её геометрический смысл

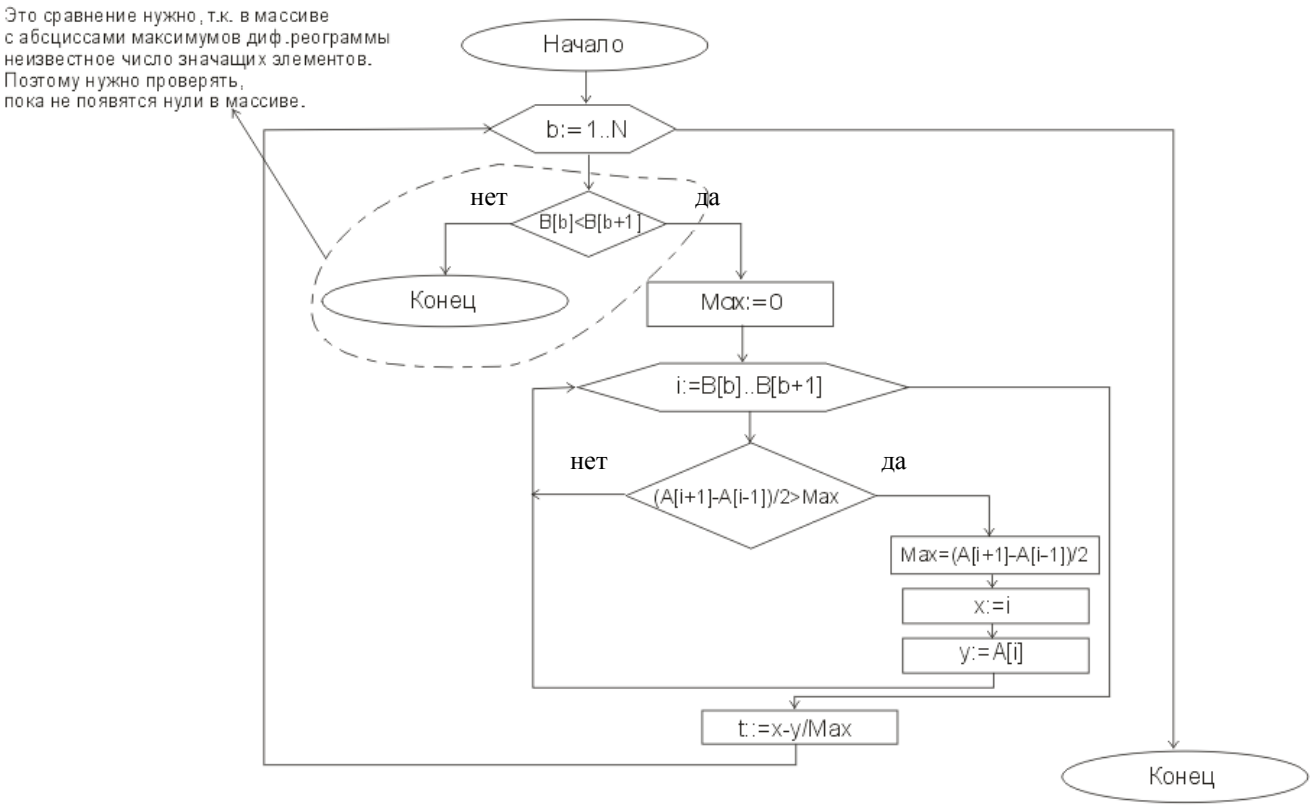

Рис. 14. Блок-схема для алгоритма нахождения точек начала изгнания крови из левого желудочка сердца.

1.11 Работа с канвой принтера

Для работой с принтером необходимо подключить модуль Printers. Начало и окончание печати сигнализируются операторами BeginDoc и EndDoc.

Для вывода текста применяются метод канвы принтера TextOut(X,Y:integer; Текст:string). Для печати графиков применяются методы MoveTo и LineTo. Их действие описано в п.1.8.

Пересчёт координат с экрана на лист осуществляется с помощью специальных функций.

Result:=GetDeviceCaps(Printer.Canvas.Handle, VertRes) – получение высоты страницы в пикселях, заданной в свойствах принтера. Метод Handle берёт эту информацию из API Windows. Аналогично действует получение ширины страницы, но вместо параметра VertRes используется HorzRes.

Result:=round(GetDeviceCaps(printer.Handle,LogPixelsX)/25.4*x) – преобразование координат по оси X с экрана на бумагу по формуле: $X_{бум} = PPI \cdot X_{экр}$, где PPI – количество пикселей на дюйм. Т.к. в русской системе предпочтительнее использовать миллиметры, то значение PPI надо разделить на количество миллиметров в дюйме – 25.4 мм. LogPixelsX возвращает PPI по оси X. Аналогично происходит пересчёт координат заменой X на Y.

Заключение

Программа имеет вид, представленный на рисунках 16-18.

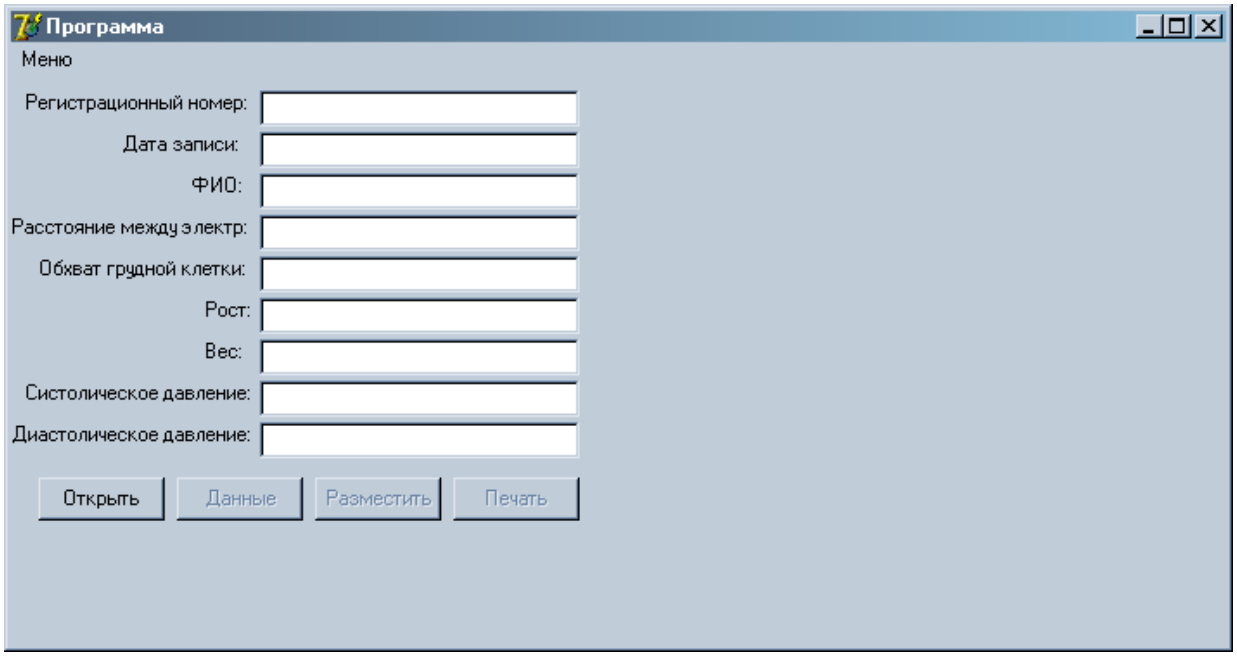

Рис. 15. Окно программы до открытия файла.

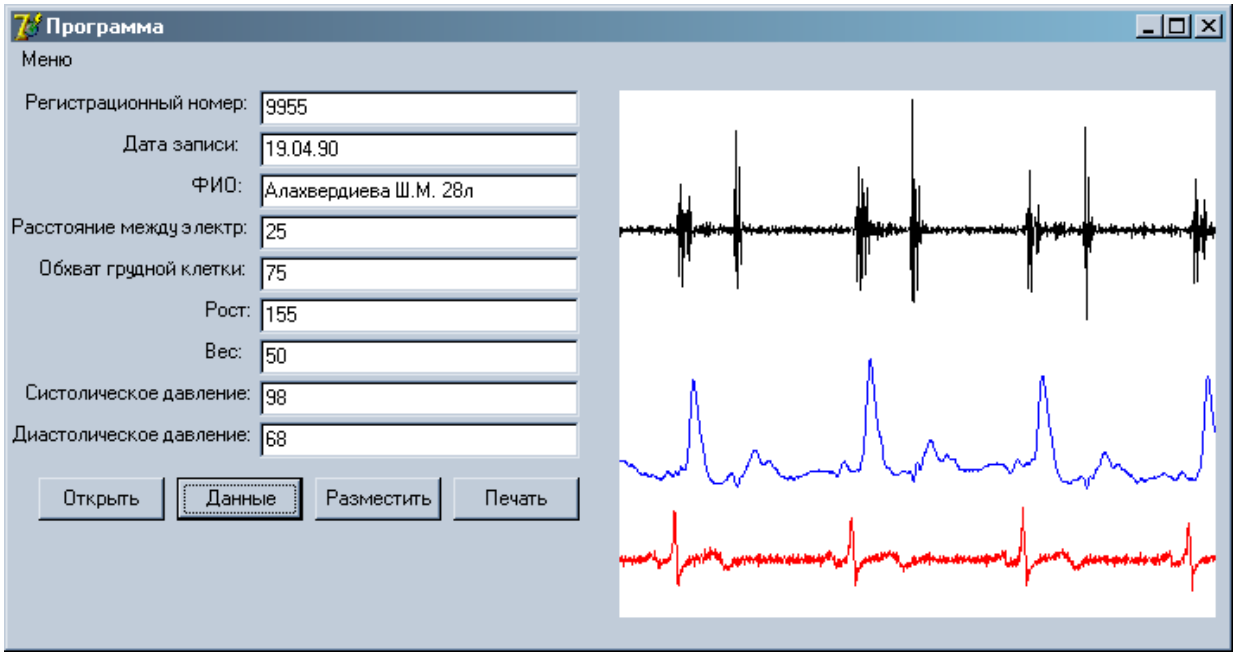

Рис. 16. Окно программы после нажатия на кнопку Данные

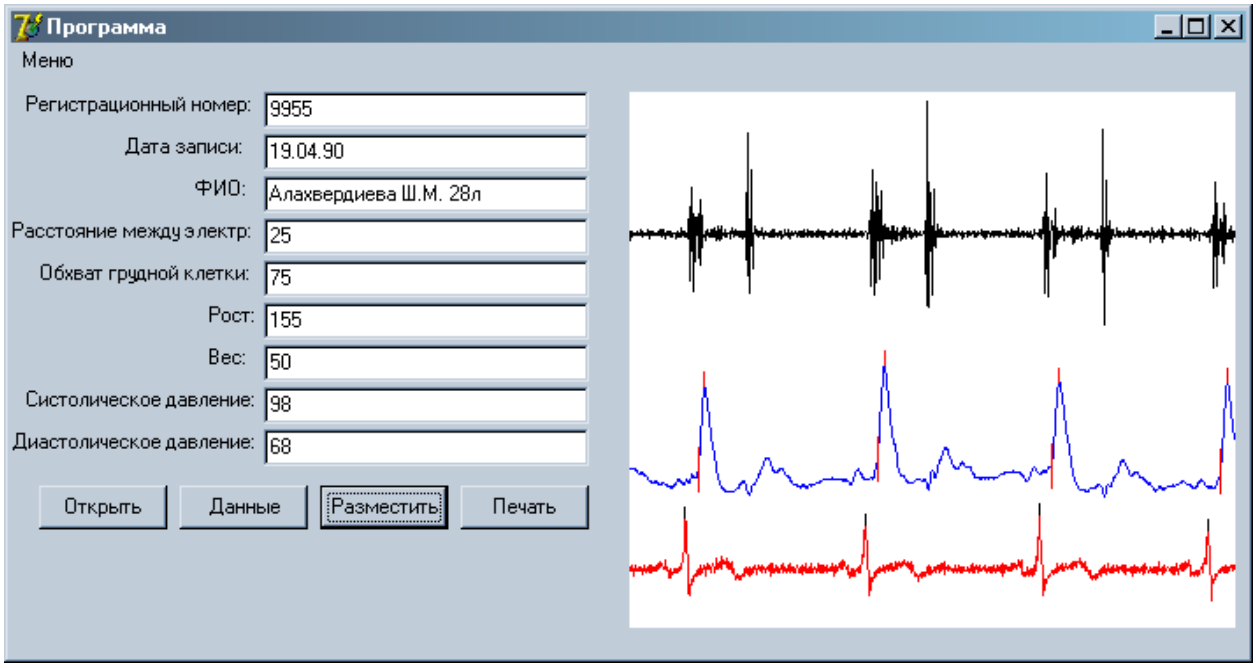

Рис. 17. Окно программы после нажатия на кнопку Разместить

Приложение А. Листинг основной программы.

```
unit Unit1;

interface

uses
  Windows, Classes, ActnList, Dialogs, Menus, StdCtrls, Controls, ExtCtrls,
  Printers, Messages, SysUtils, Variants, Graphics, Forms,
  Buttons, Grids;

type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    PopupMenu1: TPopupMenu;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    Edit5: TEdit;
    Edit6: TEdit;
    Edit7: TEdit;
    Edit8: TEdit;
    Edit9: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    OpenDialog1: TOpenDialog;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    N1: TMenuItem;
    N2: TMenuItem;
    N3: TMenuItem;
    N4: TMenuItem;
    N5: TMenuItem;
    N6: TMenuItem;
    N7: TMenuItem;
    N8: TMenuItem;
    N9: TMenuItem;
    ActionList1: TActionList;
    Action1: TAction;
    Image1: TImage;
    Action2: TAction;
    Action3: TAction;
    Action4: TAction;
```

```
    procedure Action1Execute(Sender: TObject);
    procedure Action2Execute(Sender: TObject);
    procedure Action3Execute(Sender: TObject);
    procedure Action4Execute(Sender: TObject);
    procedure MaxR;
    procedure MaxM;
    procedure GelZ;
    procedure PrintGrafik;
    procedure PrintZ;
    procedure PrintR;
    procedure PrintM;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

    function WidthX: Integer;
    function HeightY: Integer;
    function II(x:smallint):integer;
    function JJ(y,y1,y2:smallint):integer;

const
  Predel=7000;

type TReo=record
  RegNumber:smallint;
  Da_ta,Name:shortstring;
  Distance,Perimetr:smallint;
  Leng_s,Weight:smallint;
  Sist_BP,Diast_BP:smallint;
  B:array[1..4,1..Predel] of smallint;
  Kalibr_Z0,Kalibr_DZ:Real48;
  end;

var
  Form1: TForm1;
  Reo1: TReo;
  F:file of TReo;
  I1,I2,J1,J2:integer;
  ARR:array[1..80] of smallint;
  h1,h2,h3:smallint;

implementation


{$R *.dfm}

////процедура при нажатии на кнопке Открыть
procedure TForm1.Action1Execute(Sender: TObject);
begin
  OpenDialog1.Filter:='Файлы данных (*.dat)|*.dat|Все файлы|*.*';
  OpenDialog1.FilterIndex:=1;
  OpenDialog1.Title:='Выбор файла';
```

```
  if OpenDialog1.Execute then
   begin
    try
     AssignFile(F,OpenDialog1.FileName);          ////Открытие файла
     Reset(F);
     Read(F,Reo1);                                ////Чтение данных из файла в переменную
     CloseFile(F);                                ////Закрытие файла
     Button2.Enabled:=True;
     Button3.Enabled:=True;
     Button4.Enabled:=True;
     N3.Enabled:=True;
     N4.Enabled:=True;
     N5.Enabled:=True;
     N7.Enabled:=True;
     N8.Enabled:=True;
     N9.Enabled:=True;
    except                                        ////Обработка исключительных ситуаций
     on E:EInOutError do ShowMessage(E.Message);
    end;
   end;
end;

////Нахождение минимального значения ЭКГ
function MinEKG:smallint;
var
  j:integer;
begin
  Result:=Reo1.B[1,1];
  for j:=1 to Predel do
  if Reo1.B[1,j]<Result then Result:=Reo1.B[1,j];
end;

////Нахождение максимального значения ЭКГ
function MaxEKG:smallint;
var
  j:integer;
begin
  Result:=Reo1.B[1,1];
  for j:=1 to Predel do
  if Reo1.B[1,j]>Result then Result:=Reo1.B[1,j]
end;

////Нахождение минимального значения дифференцированной реограммы
function MinReo:smallint;
var
  j:integer;
begin
  Result:=Reo1.B[2,1];
  for j:=1 to Predel do
  if Reo1.B[2,j]<Result then Result:=Reo1.B[2,j];
end;

////Нахождение максимального значения дифференцированной реограммы
function MaxReo:smallint;
```

```
var
  j:integer;
begin
  Result:=Reo1.B[2,1];
  for j:=1 to Predel do
  if Reo1.B[2,j]>Result then Result:=Reo1.B[2,j]
end;

////Нахождение минимального значения ФКГ
function MinFKG:smallint;
var
  j:integer;
begin
  Result:=Reo1.B[3,1];
  for j:=1 to Predel do
  if Reo1.B[3,j]<Result then Result:=Reo1.B[3,j];
end;

////Нахождение максимального значения ФКГ
function MaxFKG:smallint;
var
  j:integer;
begin
  Result:=Reo1.B[3,1];
  for j:=1 to Predel do
  if Reo1.B[3,j]>Result then Result:=Reo1.B[3,j]
end;

////функция пересчёта координат по оси X
function II(x:smallint):integer;
var
  x1,x2:integer;
begin
  x1:=1;
  x2:=1000;
  II:=I1+Trunc((x-x1)*(I2-I1)/(x2-x1));
end;

////функция пересчёта координат по оси Y
function JJ(y,y1,y2:smallint):integer;
begin
  JJ:=J2+Trunc((y-y1)*(J1-J2)/(y2-y1));
end;

////процедура при нажатии на кнопке Данные - ввод информации в поля TEdit и
построение графика
procedure TForm1.Action2Execute(Sender: TObject);
var
  j:integer;
  A:array[0..30] of char;
begin
  Image1.Picture:=nil;                    ////Очистка TImage1
  Edit1.Text:=IntToStr(Reo1.RegNumber);    ////Ввод данных в TEdit
  Edit2.Text:=Reo1.Da_ta;
```

```
  OemToChar(StrPCopy(A,Reo1.Name),A);       ////Перекодировка строки из DOS в
Win-1251
  Edit3.Text:=A;
  Edit4.Text:=IntToStr(Reo1.Distance);
  Edit5.Text:=IntToStr(Reo1.Perimetr);
  Edit6.Text:=IntToStr(Reo1.Leng_s);
  Edit7.Text:=IntToStr(Reo1.Weight);
  Edit8.Text:=IntToStr(Reo1.Sist_BP);
  Edit9.Text:=IntToStr(Reo1.Diast_BP);
  I1:=0;                          ////Ввод координат, в пределах которых будет происходить
построение графиков
  I2:=Image1.Width;
  J1:=0;
  J2:=Round(Image1.Height)-30;

h1:=Abs(MaxEKG)+Abs(MinEKG)+Abs(MaxFKG)+Abs(MinFKG)+Abs(MaxReo)+Abs(MinR
eo);
  h2:=Abs(MaxEKG)+Abs(MinEKG)+Abs(MinReo);
  h3:=Abs(MaxEKG)+Abs(MinEKG)+Abs(MaxReo)+Abs(MinReo)+Abs(MinFKG);
  Image1.Canvas.Pen.Color:=clRed;
  Image1.Canvas.MoveTo(II(1),JJ(Reo1.B[1,1],0,h1));
  for j:=1 to 1000 do                    ////Построение графиков
    begin
      Image1.Canvas.LineTo(II(j+1),JJ(Reo1.B[1,j+1],0,h1));
    end;
    Image1.Canvas.Pen.Color:=clBlue;
    Image1.Canvas.MoveTo(II(1),JJ(Reo1.B[2,1]+h2,0,h1)+5);
  for j:=1 to 1000 do
    begin
      Image1.Canvas.LineTo(II(j+1),JJ(Reo1.B[2,j+1]+h2,0,h1)+5);
    end;
    Image1.Canvas.Pen.Color:=clBlack;
    Image1.Canvas.MoveTo(II(1),JJ(Reo1.B[3,1]+h3,0,h1)+5);
    for j:=1 to 1000 do
    begin
      Image1.Canvas.LineTo(II(j+1),JJ(Reo1.B[3,j+1]+h3,0,h1)+5);
    end;
end;

////нахождение порогового значения (для поиска R-зубцов)
function PorogR:smallint;
var
  j:integer;
  sum,max:smallint;
begin
  sum:=0;
  for j:=1 to Predel do
    sum:=sum+Reo1.B[1,j];
  max:=Reo1.B[1,1];
  for j:=1 to Predel do
    if Reo1.B[1,j]>max then max:=Reo1.B[1,j];
  Result:=Round(sum/Predel+max/2);
end;
```

```
////нахождение порогового значения (для поиска M-зубцов)
function PorogM:smallint;
var
  j:integer;
  sum,max:smallint;
begin
  sum:=0;
  for j:=1 to Predel do
   sum:=sum+Reo1.B[2,j];
  max:=Reo1.B[2,1];
  for j:=1 to Predel do
   if Reo1.B[2,j]>max then max:=Reo1.B[2,j];
  Result:=Round(sum/Predel+max/2);
end;

////нахождение R-зубцов
procedure TForm1.MaxR;
var
  j,Rx,Ry,p,p1,p2,k:integer;
begin
  Image1.Canvas.Brush.Style:=bsClear;
  Image1.Canvas.Font.Color:=clBlack;
  Image1.Canvas.Pen.Color:=clBlack;
  for j:=1 to 1000 do
    if Reo1.B[1,j]<PorogR
    then
     begin
      if PorogR<Reo1.B[1,j+1]
      then
        begin
          p1:=j+1;
          for k:=p1 to 1000 do
           if Reo1.B[1,k]>PorogR
           then
             if Reo1.B[1,k+1]<PorogR
             then
               begin
                 p2:=k;
                 Rx:=p1;
                 Ry:=Reo1.B[1,p1];
                 for p:=p1 to p2 do
                  if Reo1.B[1,p]>Ry
                  then begin Ry:=Reo1.B[1,p]; Rx:=p; end;
                 Image1.Canvas.MoveTo(II(Rx),JJ(Ry,0,h1)-7);
                 Image1.Canvas.LineTo(II(Rx),JJ(Ry,0,h1));
               end;
       end;
     end;
end;

////нахождение максимумов дифференцированной реограммы
procedure TForm1.MaxM;
var
  a,i,Mx,My,r,r1,r2,m:integer;
```

```
begin
  Image1.Canvas.Brush.Style:=bsClear;
  Image1.Canvas.Pen.Color:=clRed;
  ARR[1]:=1;
  a:=2;
  for i:=1 to 1000 do
   if Reo1.B[2,i]<PorogM
   then
    begin
     if PorogM<=Reo1.B[2,i+1]
     then
      begin
       r1:=i+1;
       for m:=r1 to 1000 do
        if Reo1.B[2,m]>=PorogM
        then
         if Reo1.B[2,m+1]<PorogM
         then
          begin
           r2:=m;
           Mx:=r1;
           My:=Reo1.B[2,r1];
           for r:=r1 to r2 do
            if Reo1.B[2,r]>My
            then begin My:=Reo1.B[2,r]; Mx:=r; end;
           ARR[a]:=Mx; a:=a+1;                    ////заполнение массива координатами точек
максимума дифференцированной реограммы по оси X (это нужно для нахождения точек
выброса крови из левого жедудочка)
           Image1.Canvas.MoveTo(II(Mx),JJ(My+h2,0,h1)+5);
           Image1.Canvas.LineTo(II(Mx),JJ(My+h2,0,h1)-4);
           Break;
          end;
      end;
    end;
end;

////Нахождение точек выбрасывания крови из левого желудочка сердца
procedure TForm1.GelZ;
var
  b,c,Px,Py,t:integer;
  Max:real;
begin
  Image1.Canvas.Brush.Style:=bsClear;
  Image1.Canvas.Pen.Color:=clRed;
  Px:=1;
  Py:=1;
  for b:=1 to 80 do
   if ARR[b]<ARR[b+1]
   then
    begin
     Max:=0.00001;
     for c:=ARR[b] to ARR[b+1] do
      begin
       if (Reo1.B[2,c+1]-Reo1.B[2,c-1])/2>Max
```

```
        then
          begin
           Max:=(Reo1.B[2,c+1]-Reo1.B[2,c-1])/2;
           Px:=c;
           Py:=Reo1.B[2,c];
          end;
       end;
      t:=Round(Px-Py/Max);
      Image1.Canvas.MoveTo(II(t),JJ(Reo1.B[2,t]+h2,0,h1)+20);
      Image1.Canvas.LineTo(II(t),JJ(Reo1.B[2,t]+h2,0,h1)-6);
     end
    else Break;
end;

////процедура при нажатии на кнопку Разместить
procedure TForm1.Action3Execute(Sender: TObject);
begin
  MaxR;
  MaxM;
  GelZ;
end;

////функция получения высотры страницы в пикселях
function HeightY: Integer;
begin
  Result:=GetDeviceCaps(Printer.Canvas.Handle, VertRes)
end;

////функция получения ширины страницы в пикселях
function WidthX: Integer;
begin
  Result:=GetDeviceCaps(Printer.Canvas.Handle, HorzRes)
end;

////функция преобразования координат с бумаги по оси X
function PrinterCoordX(x:integer):integer;
begin
  Result:=round(GetDeviceCaps(printer.Handle,LogPixelsX)/25.4*x);
end;

////функция преобразования координат с бумаги по оси Y
function PrinterCoordY(Y:integer):integer;
begin
  Result:=round(GetDeviceCaps(printer.Handle,LogPixelsY)/25.4*Y);
end;

////печать графиков
procedure TForm1.PrintGrafik;
var
  j:integer;
begin
  I1:=0;
  I2:=Round(WidthX);
  J1:=0;
```

```
  J2:=Round(HeightY/2);
  Printer.Canvas.MoveTo(II(1),PrinterCoordY(10+9*Font.Size)+JJ(Reo1.B[1,1],0,h1));
  for j:=1 to 1000 do
   begin
    Printer.Canvas.LineTo(II(j+1),PrinterCoordY(10+9*Font.Size)+JJ(Reo1.B[1,j+1],0,h1));
   end;
  Printer.Canvas.MoveTo(II(1),PrinterCoordY(10+9*Font.Size)+JJ(Reo1.B[2,1]+h2,0,h1)+5);
  for j:=1 to 1000 do
   begin

Printer.Canvas.LineTo(II(j+1),PrinterCoordY(10+9*Font.Size)+JJ(Reo1.B[2,j+1]+h2,0,h1)+5);
   end;
  Printer.Canvas.MoveTo(II(1),PrinterCoordY(10+9*Font.Size)+JJ(Reo1.B[3,1]+h3,0,h1)+5);
  for j:=1 to 1000 do
   begin

Printer.Canvas.LineTo(II(j+1),PrinterCoordY(10+9*Font.Size)+JJ(Reo1.B[3,j+1]+h3,0,h1)+5);
   end;
end;

////печать R-зубцов
procedure TForm1.PrintR;
var
  j,Rx,Ry,p,p1,p2,k:integer;
begin
  Printer.Canvas.Brush.Style:=bsClear;
  I1:=0;
  I2:=Round(WidthX);
  J1:=0;
  J2:=Round(HeightY/2);
  for j:=1 to 1000 do
   if Reo1.B[1,j]<PorogR
   then
    begin
     if PorogR<Reo1.B[1,j+1]
     then
      begin
       p1:=j+1;
       for k:=p1 to 1000 do
        if Reo1.B[1,k]>PorogR
        then
          if Reo1.B[1,k+1]<PorogR
          then
           begin
            p2:=k;
            Rx:=p1;
            Ry:=Reo1.B[1,p1];
            for p:=p1 to p2 do
             if Reo1.B[1,p]>Ry
             then begin Ry:=Reo1.B[1,p]; Rx:=p; end;
            Printer.Canvas.MoveTo(II(Rx),PrinterCoordY(10+9*Font.Size)+JJ(Ry,0,h1)-20);
            Printer.Canvas.LineTo(II(Rx),PrinterCoordY(10+9*Font.Size)+JJ(Ry,0,h1)+30);
           end;
      end;
```

```
     end;
end;

////печать точек выброса крови из левого желудочка
procedure TForm1.PrintZ;
var
  b,c,Px,Py,t:integer;
  Max:real;
begin
  Printer.Canvas.Brush.Style:=bsClear;
  I1:=0;
  I2:=Round(WidthX);
  J1:=0;
  J2:=Round(HeightY/2);
  Px:=1;
  Py:=1;
  for b:=1 to 80 do
   if ARR[b]<ARR[b+1]
   then
    begin
     Max:=1;
     for c:=ARR[b] to ARR[b+1] do
      begin
       if (Reo1.B[2,c+1]-Reo1.B[2,c-1])/2>Max
       then
        begin
         Max:=(Reo1.B[2,c+1]-Reo1.B[2,c-1])/2;
         Px:=c;
         Py:=Reo1.B[2,c];
        end;
      end;
     t:=Round(Px-Py/Max);
     Printer.Canvas.MoveTo(II(t),PrinterCoordY(10+9*Font.Size)+JJ(Reo1.B[2,t]+h2,0,h1)-40);
     Printer.Canvas.LineTo(II(t),PrinterCoordY(10+9*Font.Size)+JJ(Reo1.B[2,t]+h2,0,h1)+40);
    end
   else Break;
end;

////печать максимумов дифференцированной реограммы
procedure TForm1.PrintM;
var
  a,i,Mx,My,r,r1,r2,m:integer;
begin
  Printer.Canvas.Brush.Style:=bsClear;
  I1:=0;
  I2:=Round(WidthX);
  J1:=0;
  J2:=Round(HeightY/2);
  ARR[1]:=1;
  a:=2;
  for i:=1 to 1000 do
   if Reo1.B[2,i]<PorogM
   then
    begin
```

```
      if PorogM<=Reo1.B[2,i+1]
      then
       begin
        r1:=i+1;
        for m:=r1 to 1000 do
         if Reo1.B[2,m]>=PorogM
         then
          if Reo1.B[2,m+1]<PorogM
          then
           begin
            r2:=m;
            Mx:=r1;
            My:=Reo1.B[2,r1];
            for r:=r1 to r2 do
            if Reo1.B[2,r]>My
             then begin My:=Reo1.B[2,r]; Mx:=r; end;
            ARR[a]:=Mx; a:=a+1;
            Printer.Canvas.MoveTo(II(Mx),PrinterCoordY(10+9*Font.Size)+JJ(My+h2,0,h1)-20);
            Printer.Canvas.LineTo(II(Mx),PrinterCoordY(10+9*Font.Size)+JJ(My+h2,0,h1)+30);
            Break;
           end;
       end;
     end;
end;

////процедура при нажатии на кнопку Печать
procedure TForm1.Action4Execute(Sender: TObject);
begin
  Printer.BeginDoc;
  Printer.Canvas.TextOut(PrinterCoordX(10), PrinterCoordY(10),Label1.Caption+'
'+Edit1.Text);
  Printer.Canvas.TextOut(PrinterCoordX(10),
PrinterCoordY(10+Round(Font.Size/2)),Label2.Caption+' '+Edit2.Text);
  Printer.Canvas.TextOut(PrinterCoordX(10),
PrinterCoordY(10+Round(2*Font.Size/2)),Label3.Caption+' '+Edit3.Text);
  Printer.Canvas.TextOut(PrinterCoordX(10),
PrinterCoordY(10+Round(3*Font.Size/2)),Label4.Caption+' '+Edit4.Text);
  Printer.Canvas.TextOut(PrinterCoordX(10),
PrinterCoordY(10+Round(4*Font.Size/2)),Label5.Caption+' '+Edit5.Text);
  Printer.Canvas.TextOut(PrinterCoordX(10),
PrinterCoordY(10+Round(5*Font.Size/2)),Label6.Caption+' '+Edit6.Text);
  Printer.Canvas.TextOut(PrinterCoordX(10),
PrinterCoordY(10+Round(6*Font.Size/2)),Label7.Caption+' '+Edit7.Text);
  Printer.Canvas.TextOut(PrinterCoordX(10),
PrinterCoordY(10+Round(7*Font.Size/2)),Label8.Caption+' '+Edit8.Text);
  Printer.Canvas.TextOut(PrinterCoordX(10),
PrinterCoordY(10+Round(8*Font.Size/2)),Label9.Caption+' '+Edit9.Text);
  PrintGrafik;
  PrintR;
  PrintM;
  PrintZ;
  Printer.EndDoc;
end;
```

end.

Приложение Б. Вывод информации на печать.

Список литературы

1. Цегельникова А.Л. Методические указания по выполнению лабораторных работ. Реография. Обработка сигнала и расчёт гемодинамических показателей в системе Delphi. – М.:МИРЭА, 2004. – 20с.
2. http://lainslav.narod.ru/ - Медицина – Реография.