Языки программирования — задания с использованием библиотек

1. Общие сведения

Поскольку сборка и подключение библиотек в ОС Windows часто требует дополнительных и нестандартных усилий, в том числе, зависящих от используемого компилятора, если вообще возможно, основной средой для решения задачи 8 является предоставленная в ваше пользование виртуальная машина с ОС Linux. В силу того, что используемый дистрибутив предполагает сборку всей системы из исходных текстов, наличие работающей графической среды и прикладных программ уже подразумевает наличие множества установленных библиотек, готовых для использования. Тем не менее, в рамках этой работы требуется продемонстрировать умение устанавливать библиотеки вручную стандартным способом, в обход специфического для каждого дистрибутива Linux менеджера пакетов.

В каждом варианте будет указано, установлена ли требуемая библиотека в имеющейся системе, и, если нет, приведена команда по её установки средствами менеджера пакетов дистрибутива Gentoo. Это команда должна быть выполнена при наличии подключения к интернету от имени пользователя root после того, как вы восстановите базу данных менеджера пакетов (см. файл ПРОЧТИ МЕНЯ на рабочем столе виртуальной машины). При этом библиотека будет установлена в стандартные каталоги операционной системы (дерево `/usr`), которые входят в пути поиска заголовочных файлов и библиотек компилятора и компоновщика по умолчанию. Узнать список всех файлов, входящих в установленный пакет, можно командой `equery files имя-пакета` (если пакет уже установлен).

Вам останется только включить требуемые заголовочные файлы директивами `#include <...>` и указать имена требуемых библиотек. Имена заголовочных файлов обычно указаны в документации на соответствующие функции или классы библиотеки. Иногда заголовочные файлы и библиотеки расположены в подкаталогах системных каталогов, которые потребуется добавить, как описано ниже.

Библиотеки указываются в файле проекта с помощью строки `LIBS += имена библиотек`. Указывать в этом списке следует только базовые имена библиотек без префикса `lib` и расширений, включая номера версий, добавив перед ними слитно префикс `-l` (таков формат опций драйвера транслятора), например, для подключения библиотеки из файла `libstuff.so.0.0.0` следует указать `-lstuff`. Имя библиотеки можно уточнить исходя из списка файлов пакета.

Для некоторых библиотек заголовочные файлы или каталоги с ними, указанные в документации, находятся не непосредственно в основном системном каталоге для них `/usr/include`, а в его подкаталогах. Для таких библиотек даже при использовании системной копии необходимо указывать пути поиска заголовочных файлов, как описано ниже.

После того, как вы убедитесь, что ваша программа компонуется и работает с требуемой библиотекой корректно, вам потребуется установить ещё одну копию библиотеки самостоятельно, независимыми от дистрибутива средствами, в отдельное дерево каталогов, например, `/home/user/prefix`, не связанное с системными каталогами. Для работы с такой библиотекой необходимо добавить в файл проекта следующие записи:

- `INCLUDEPATH += /home/user/prefix/include` — добавление требуемых каталогов в путь поиска заголовочных файлов директивой `#include` в форме с угловыми скобками. Для UNIX-компиляторов соответствует опции `-I`.
- `LIBS += -L/home/user/prefix/lib` — добавление требуемых каталогов в путь поиска библиотек в процессе компоновки. (имена библиотек тоже добавляются к переменной LIBS, как было показано выше, сначала указываются пути поиска, затем — сами библиотеки).
- `QMAKE_RPATHDIR += /home/user/prefix/lib` — добавление требуемых каталогов в путь поиска библиотек в процессе динамической компоновки при запуске программы. Для UNIX-компиляторов соответствует опции `-Wl,-rpath`. Для Windows не работает и не требуется, т.к. там включение путей поиска зависимых библиотек в саму библиотеку не поддерживается на уровне ОС.

Некоторые библиотеки для C++ состоят только из определений шаблонов и встраиваемых функций в заголовочных файлах, для них необходимо указать только пути их поиска. Такие библиотеки обычно не требуют никакой процедуры сборки.

Чтобы убедиться, что ваша программа скомпоновалась с вашей версией библиотеки, можно воспользоваться командой `lddtree имя-исполняемого-файла`, которая показывает дерево зависимостей динамических библиотек — она должна показать полные пути к динамическим библиотекам, которые будут загружены при запуске программы.

2. Стандартная сборка пакетов

Распаковать скачанный файл в формате `.tar.gz/bz2/xz/lz` можно командой `tar xvf имя-файла-архива`. Процесс сборки пакета обычно состоит из трёх этапов, команды которых необходимо выполнить в каталоге, куда вы распаковали архив:

1. Конфигурация. На этом этапе специальный скрипт, являющийся частью системы сборки, анализирует среду выполнения, для которой будет производиться сборка. В эти свойства входят, например, характеристики реализации языка C в части зависящего от реализации поведения, а также поиск других библиотек, которые необходимы или могут опционально использоваться данной. Иногда этот этап отсутствует. Большинство пакетов используют для конфигурации систему autotools, скрипт конфигурации которой называется `configure`. Другие системы сборки часто совместимы с ним. Вам потребуется выполнить следующую команду для конфигурации:

   ```
   CC=clang CXX=clang++ ./configure --prefix=/home/user/prefix --enable-shared --disable-static
   ```

   Здесь через переменные окружения указано использование компилятора clang вместо системного, а ключами задано использование дерева для установки («префикса») и сборка только динамических библиотек.

2. Сборка. При использовании системы сборки `make` сборка осуществляется одноимённой командой. Для ускорения сборки можно указать этой команде ключ `-jN`, где N — число параллельно запускаемых задач сборки, обычно устанавливаемое в число ядер процессора.

3. Установка результатов сборки в префикс. При использовании системы сборки `make` выполняется командой `make install`. Убедитесь, что после её завершения в подкаталогах префикса `include` содержатся заголовочные файлы библиотеки, а в подкаталоге `lib` — динамические библиотеки с расширением `.so`.

Для библиотек, которые требуют других шагов сборки, они будут указаны отдельно. Выполнять их нужно в каталоге, куда были распакованы исходные тексты библиотеки.

3. Общие рекомендации

- Для запроса имён обрабатываемых файлов у пользователя используйте метод `QFileDialog::getOpenFileName`, имён каталогов — `QFileDialog::getExistingDirectory`. Для запроса имён ещё, скорее всего, не существующих выходных файлов используйте метод `QFileDialog::getSaveFileName`.
- Для доступа к файлам используйте класс `QFile`.
- Для выполнения периодических действий по таймеру используйте класс `QTimer`.
- В задачах обработки файлов вам необходимо выбрать размер блока, например, 1 мегабайт, и обрабатывать файл по частям, поскольку легко представить файл, который не поместится в ОЗУ целиком. При этом вам потребуется особо обработать ситуацию, в которой размер входного файла не кратен размеру блока.

4. Задачи

4.1. gmp

Сайт: `http://gmplib.org`
Сборка: установлена в виртуальной машине, вручную — стандартным образом.

Задание: реализовать калькулятор, выполняющий четыре основные арифметические операции над целыми числами не ограниченной каким-либо типом ширины.

Материал:

- Общие положения о C++-интерфейсе библиотеки: `https://gmplib.org/manual/C_002b_002b-Interface-General.html`
- Дополнительная информация о классе обработки целых чисел: `https://gmplib.org/manual/C_002b_002b-Interface`
`html`

4.2. QtNetwork: сетевой чат

Сайт: `http://qt-project.org/doc/qt-5/qtnetwork-programming.html`

Сборка: не требуется.

Задание: реализовать сетевой чат между двумя экземплярами программы по протоколу TCP.

Материал:

- Класс `QTCPSocket`, являющийся объектом-соединением с другим таким же сокетом по сети: `http://qt-project.org/doc/qt-5/qtcpsocket.html`
- Класс `QTCPServer`, являющийся объектом-сервером, принимающим входящие подключение от других сокетов и создающий соединённые пары: `http://qt-project.org/doc/qt-5/qtcpserver.html`

В вашей программе с графическим интерфейсом пользоваться методами ожидания на сокетах вида `waitFor...` нельзя, поскольку они будут блокировать цикл обработки сообщений, вместо этого, вся обработка событий должна быть реализована с помощью механизма сигналов/слотов. Учтите, что, как и при чтении файлов, при чтении данных из сокета TCP границы блоков принимаемых данных не соответствуют вызовам записи передающей стороны, а могут быть любыми. Вам придётся изобрести свой протокол, использующий специальные признаки конца данных или пересылающий длину данных в начале. В обработчике сигнала `readyRead` может придти любое количество данных, если это неполный блок, вам необходимо сохранить принятые данные и ждать следующего такого сигнала.

Дополнительный вопрос: поскольку ваша задача не требует библиотеки помимо Qt, вам необходимо продемонстрировать результат успешной сборки библиотеки gmp.

4.3. libzip

Сайт: `http://www.nih.at/libzip/index.html`

Сборка: с помощью менеджера пакетов: `emerge libzip`, вручную — стандартным образом.

Задание: вывести дерево файлов, содержащихся в указанном пользователем архиве.

Материал: для поддержки не-ASCII путей используйте функцию `zip_fdopen`, передавая ей дескриптор открытого с помощью класса `QFile` файла, который можно получить через вызов `QFile::handle()`. Далее потребуется узнать число файлов в архиве через `zip_get_num_entries`, после чего информацию о каждом отдельном файле по его номеру можно получить с помощью `zip_stat_index`. Закрывается архив функцией `zip_discard`. Документацию по функциям библиотеки libzip можно получить командой `man имя-функции`.

4.4. exiv2

Сайт: `http://www.exiv2.org/`

Сборка: с помощью менеджера пакетов: `emerge exiv2`, вручную — стандартным образом.

Задание: вывести метаданные выбранного пользователем файла изображения в виде таблицы.

Материал: пример считывания данных с выводом в консоль: `http://www.exiv2.org/doc/exifprint_8cpp-example.html`. Для поддержки не-ASCII путей используйте вместе приведённого в этом примере вызова `Exiv2::ImageFactory::open` его перегрузку, принимающую блок данных и его размер. Сами данные считайте с помощью объекта класса `QFile`.

4.5. python 3

Сайт: `http://python.org`

Сборка: установлен в виртуальной машине, вручную — стандартным способом (убедитесь, что скачали python серии 3, а не 2).

Задание: выполнить скрипт, введённый в интерфейсе программы на языке python и отобразить в графмческом интерфейсе выводимый им текст.

Материал: пример встраивания интерпретатора с захватом стандартного вывода:

```
#include <python3.4/Python.h>
#include <iostream>

static PyObject* redirection_stdout_redirection(PyObject* self,PyObject* args)
{
    const char* s;
    if(!PyArg_ParseTuple(args,”s”,&s))
        return nullptr;
    std::cout << ”Output: ” << s << ’\n’;
    Py_RETURN_NONE;
}

static PyMethodDef redirection_methods[] = {
    {”stdout_redirection”,redirection_stdout_redirection,METH_VARARGS,””},
    {}
};

static PyModuleDef redirection_module = {
    PyModuleDef_HEAD_INIT,
    ”redirection”,
    ””,
    0,
    redirection_methods
};

static PyObject* PyInit_redirection()
{
    return PyModule_Create(&redirection_module);
}

int main()
{
    // Добавить модуль перехвата в список встроенных.
    PyImport_AppendInittab(”redirection”,PyInit_redirection);
    // Инициализировать интерпретатор.
    Py_Initialize();
    // Выполнить скрипт, привязывающий стандартный вывод
    // интерпретатора к перехватчику.
    PyRun_SimpleString(R”(
import io,sys,redirection

class StdoutRedirector(io.IOBase):
    write = redirection.stdout_redirection

sys.stdout = StdoutRedirector()
)”);
    // Теперь можно выполнять произвольные скрипты:
    PyRun_SimpleString(”print(’abc привет!’)”);
    Py_Finalize();
}
```

Вам потребуется разделить этот пример на инициализацию, выполняемую в начале программы, и вызовы `PyRun_SimpleString` для введённого пользователем кода. В функции `redirection_stdout_redirection` потре-

буется заменить вывод в консоль записью в текстовое поле в интерфейсе программы, вам предлагается воспользоваться виджетом `QPlainTextEdit`. Для добавления в конец текста можно сначала переместить курсор в конец методом `QPlainTextEdit::moveCursor`, а затем вызвать `QPlainTextEdot::insertPlainText`.

4.6. QtMultimedia

Сайт: `http://qt-project.org/doc/qt-5/qtmultimedia-index.html`

Сборка: не требуется.

Задание: реализуйте интересную вам игру или другое мультимедиа-приложение с воспроизведением звука и/или видео.

Материал: для воспроизведения мультимедиа используется класс `QMediaPlayer`, пример его использования дан в справке по нему. Остальная часть задания отдаётся на ваше усмотрение, будьте внимательны с расчётом собственных сил, исходя из имеющегося времени!

Дополнительный вопрос: поскольку ваша задача не требует библиотеки помимо Qt, вам необходимо продемонстрировать результат успешной сборки python.

4.7. poppler

Сайт: `http://poppler.freedesktop.org/`

Сборка: с помощью менеджера пакетов:

```
echo -e -qt5 >> /etc/portage/profile/use.mask
echo ”app-text/poppler qt5” >> /etc/portage/package.use/qt-creator
echo ”app-text/poppler @clang” >> /etc/portage/package.vars
emerge poppler
```

Вручную: слишком сложно. Вместо этого продемонстрируйте успешную сборку библиотеки exiv2.

Задание: отобразите в графическом интерфейсе первую страницу выбранного пользователем pdf-файла.

Материал: документация по интерфейсу для использования с Qt 5, включая пример: `http://people.freedesktop.org/~aacid/docs/qt5/`

Для компоновки необходимо использовать две библиотеки: `poppler-qt5` и `poppler`.

4.8. libquvi

Сайт: `http://quvi.sourceforge.net/`

Сборка: с помощью менеджера пакетов: `emerge libquvi`, вручную: слишком сложно. Вместо этого продемонстрируйте успешную сборку библиотеки gmp.

Задание: по данной ссылке на страницу с видео на сайте youtube выведите сведения о видео и ссылки для его загрузки в разных форматах.

Материал: документация по libquvi, включая примеры: `http://quvi.sourceforge.net/r/api/0.9/`. Для отображения свойств каждого потока в табличном виде можно использовать виджет `QTableWidget`.

4.9. QtNetwork: доступ к http

Сайт: `http://qt-project.org/doc/qt-5/qtnetwork-programming.html`

Сборка: не требуется.

Задание: получить прогноз погоды от веб-сервиса и отобразить в интерфейсе.

Материал: данные в формате JSON можно получить через API, описанный в `https://openmeteoforecast.org/wiki/API/0.1`. Для чтения ответов на http-запросы необходимо использовать класс `QNetworkAccessManager`, пример использования дан в его документации. Ответ в формате JSON можно разобрать с помощью метода `QJsonDocument::fromBinaryData`.

Дополнительный вопрос: поскольку ваша задача не требует библиотеки помимо Qt, вам необходимо продемонстрировать результат успешной сборки библиотеки, используемой в предыдущем задании, без её дальнейшего использования.

4.10. id3lib

Сайт: http://id3lib.sourceforge.net/
Сборка: с помощью менеджера пакетов: emerge id3lib. Вручную для Windows:

1. Поставить msys: http://sourceforge.net/projects/mingwbuilds/files/external-binary-packages/msys%2B7za%2Bwget%2Bsvn%2Bgit%2Bmercurial%2Bcvs-rev13.7z/download.
2. В файле etc\fstab указать путь к MinGW от Qt Creator с прямыми слешами, отображённый в /mingw:

   ```
   C:/Qt/Tools/mingw482_32 /mingw
   ```

3. Запустить msys.bat. Содержимое дисков ОС видно в путях, начинающихся с корня и одной строчной буквы диска, например, вместо C:\a.txt следует писать /c/a.txt.
4. Выполнить команды:

   ```
   wget http://sourceforge.net/projects/id3lib/files/id3lib/3.8.3/id3lib-3.8.3.tar.gz/download
   tar xvf id3lib-3.8.3.tar.gz
   cd id3lib-3.8.3
   wget --no-check-certificate https://raw.githubusercontent.com/arktools/mingw-cross-env/master/src/id3lib-1-win32.patch
   patch -p1 < id3lib-1-win32.patch
   autoconf
   ./configure
   make
   make install
   ```

5. Добавить пути local/include и local/lib относительно каталога установки msys в настройки проекта.

Задание: отобразить в виде таблицы теги и их значения для указанного пользователем музыкального файла.
Материал: документация с примерами: http://id3lib.sourceforge.net/api/index.html. Для отображения тегов в табличном виде можно использовать виджет QTableWidget.

4.11. graphviz

Сайт: http://www.graphviz.org
Сборка: с помощью менеджера пакетов:

```
echo ”media-gfx/graphviz @no-lto” >> /etc/portage/package.vars
emerge graphviz
```

Вручную собирать его слишком сложно, вместо этого продемонстрируйте успешную сборку библиотеки libzip.
Задание: по заданному пользователем графу построить его планарное представление.
Материал:

- Необходимо использовать graphviz в качестве библиотеки без вызова внешних программ. Исходные данные можно запрашивать в любом виде (матрица инцидентности, смежности, ...).
- Документация по библиотеке описания графов: http://www.graphviz.org/pdf/Agraph.pdf, главы 1–5. К сожалению, все строки библиотека принимают в виде параметров типа char*, т.е. даже для обычных строковых литералов требуется const_cast или другие способы для отнятия квалификатора const, хотя библиотека их менять и не собирается.
- Документация по алгоритмам раскладки графа: http://www.graphviz.org/doc/libguide/libguide.pdf.

Пример раскладки графа и извлечения информации для его отображения:

```
#include <iostream>
#include <sstream>

#include <gvc.h>
```

```
int main(int argc, char **argv)
{
    // Создать контекст раскладки графов
    GVC_t* gvc = gvContext();

    // Создать ненаправленый граф
    char graphName[] = ”G”;
    Agraph_t* graph = agopen(graphName,Agundirected,nullptr);

    // Построим полносвязный граф из n вершин
    constexpr size_t n = 5;

    // Создадим вершины
    Agnode_t* nodes[n];
    char nodeName[] = ”0”;
    for(size_t i=0;i<n;++i,++nodeName[0])
        nodes[i] = agnode(graph,nodeName,true);

    // В полносвязном графе n*(n-1)/2 дуг
    constexpr size_t ne = n*(n-1)/2;

    // Создать дуги
    Agedge_t* edges[ne];
    size_t k = 0;
    for(size_t i=0;i<n-1;++i){
        for(size_t j=i+1;j<n;++j){
            std::ostringstream oss;
            oss << i << ’-’ << ’j’;
            std::string s = oss.str();
            edges[k++] = agedge(graph,nodes[i],nodes[j],&s[0],true);
        }
    }

    // Сделаем раскладку графа. Можны предложить пользователю
    // разные алгоритмы на выбор, вот полный список:
    // circo dot fdp neato nop nop1 nop2 osage patchwork sfdp twopi
    gvLayout(gvc,graph,”circo”);

    // Для тестирования своего рендеринга можно сохранит в графический
    // файл эталонный вариант, нарисованный самой библиотекой.
    // Пользоваться им можно только для отладки, наша задача -
    // нарисовать всё самим.
    // gvRenderFilename(gvc,graph,”png”,”graph.png”);

    // У graphviz система координат с осью y не в том направлении, как в Qt,
    // нужно её обратить для всех точек, чтобы сплайны корректно рисовались.
    // Для этого нужна высота графа.

    double h = GD_bb(graph).UR.y;

    // Положение и размер вершин. Мы не задавали их форму, по умолчанию это эллипсы.
    // Координаты даны центра, а ширина и высота в дюймах, которые надо перевести
    // в пиксели.
    double dpi = 72.;
    for(size_t i=0;i<n;++i){
        const pointf& pos = ND_coord(nodes[i]);
```

```
    std::cout << "Node " << i << " at (" << pos.x << ',' << h-pos.y << "), size "
              << ND_width(nodes[i])*dpi << 'x' << ND_height(nodes[i])*dpi << '\n';
}

// Дуги
for(size_t i=0;i<ne;++i){
    std::cout << "Edge " << i << ":\n";
    // Дуга состоит из нескольких сплайнов (обычно одного)
    const splines* s = ED_spl(edges[i]);
    for(size_t j=0;j<s->size;++j){
        std::cout << "    spline " << j << ":\n";
        const bezier& b = s->list[j];
        // Если есть начальный сегмент, перейдём в него и нарисуем линию до нулевой
        // точки кривых.
        if(b.sflag){
            std::cout << "        move to (" << b.sp.x << ',' << h-b.sp.y << ")\n";
            std::cout << "        line to (" << b.list[0].x << ',' << h-b.list[0].y << ")\n";
        }else
        // Если начального сегмента нет, просто переходим в начальную точку.
            std::cout << "        move to (" << b.list[0].x << ',' << h-b.list[0].y << ")\n";
        // После начальной точки идут тройки из двух контролирующих точек кривой Безье
        // и конечной.
        for(size_t k=1;k<b.size;k+=3){
            std::cout << "        bezier to";
            for(size_t l=0;l<3;++l)
                std::cout << " (" << b.list[k+l].x << ',' << h-b.list[k+l].y << ')';
            std::cout << '\n';
        }
        // Если есть конечный сегмент, дорисуем прямую до него.
        if(b.eflag)
            std::cout << "        line to (" << b.ep.x << ',' << h-b.ep.y << ")\n";
    }
}

// Освободим все ресурсы.
gvFreeLayout(gvc,graph);
for(size_t i=0;i<ne;++i)
    agdeledge(graph,edges[i]);
for(size_t i=0;i<n;++i)
    agdelnode(graph,nodes[i]);
agclose(graph);
gvFreeContext(gvc);
```

Для компоновки необходимо использовать три библиотеки: `gvc`, `cgraph` и `cdt`.

4.12. libgcrypt

Сайт: `http://www.gnu.org/software/libgcrypt/`

Сборка: установлена в виртуальной машине, вручную — стандартным способом.

Задание: вычислите значения криптографической функции ГОСТ 34.11-2012 для всех файл в указанном пользователем каталоге и отобразите в виде таблицы.

Материалы:

- Инициализация библиотеки: `https://www.gnupg.org/documentation/manuals/gcrypt/Initializing-the-library.html`
- Интерфейс хеш-функций: `https://www.gnupg.org/documentation/manuals/gcrypt/Working-with-hash-algorithm.html`

- 512-битная версия ГОСТ 34.11-2012 имеет в этой библиотеке идентификатор `GCRY_MD_STRIBOG512`.
- Имена всех файлов в каталоге можно узнать с помощью функции `QDir::entryList`.

Для компоновки необходимо использовать две библиотеки: `gcrypt` и `gpg-error`.

4.13. bzip2

Сайт: `http://http://www.bzip.org/`

Сборка: установлена в виртуальной машине, вручную — слишком сложно. Вместо этого продемонстрируйте успешную сборку библиотеки libgcrypt.

Задание: Распакуйте или запакуйте указанный пользователем файл с помошью библиотеки bzip2.

Материал: используйте низкоуровнёвый интерфейс, описанный в `http://www.bzip.org/1.0.5/bzip2-manual-1.0.5.html` для поблочной обработки файлов, открытых с помощью объекта класса `QFile`.

4.14. mpfr

Сайт: `http://http://www.mpfr.org/`

Сборка: установлена в виртуальной машине, вручную — стандартным способом.

Задание: вычислите значения функций синуса и косинуса для данного значения угла в градусах с произвольной точностью в битах, указанной пользователем. Выведите результаты и графическое представление в виде диаграммы с единичной окружностью, лучом, и высотами, опущенными на оси координат.

Материал: вычисление можно провести с помощью разложения в ряд Маклорена, а входящие в него арифметические операции выполнить с помощью библиотеки. Документация: `http://www.mpfr.org/mpfr-current/mpfr.html#MPFR-Interface`.

4.15. libstatgrab

Сайт: `http://www.i-scream.org/libstatgrab/`

Сборка: с помощью менеджера пакетов: `emerge libstatgrab`, вручную — стандартным образом.

Задание: выведите и обновляйте в реальном времени индикаторы загруженности процессора, использования памяти, диска и сети.

Материалы: инициализация и завершение работы с библиотекой осуществляется функциями `sg_init` и `sg_shutdown` соответственно. Необходимую статистку можно получить вызовами `sg_get_cpu_percents`, `sg_get_mem_stats`, `sg_get_disk_io_stats_diff` и `sg_get_network_io_stats_diff`.

4.16. boost.math

Сайт: `http://boost.org`

Сборка: не требуется, заголовочные файлы уже установлены в виртуальной машине.

Задание: выведите график Дзета-функции Римана. Реализуйте возможность перемещаться по плоскости для осмотра разных частей графика перетаскиванием мышью, а также управление масштабом с помощью колеса мыши. Кроме этого реализуйте ручной ввод координат области координатной плоскости, отображаемой на экране.

Материалы: функция вычисляется с помощью указанной библиотеки, документация по требуемой функции: `http://www.boost.org/doc/libs/1_57_0/libs/math/doc/html/math_toolkit/zetas/zeta.html`.

Дополнительный вопрос: поскольку ваша задача не требует сборки библиотеки, вам необходимо продемонстрировать результат успешной сборки библиотеки libstatgrab.

4.17. Drag & Drop

Сайт: `http://qt-project.org/doc/qt-5/dnd.html`

Сборка: не требуется.

Задание: разберитесь с механизмом перетаскивания данных библиотеки Qt. Ваша программа должна уметь принимать на себя объекты, перетаскиваемые мышью из других программ (если они на это рассчитаны) и отображать о них всю возможную информацию. Сделайте то же самое для текущего содержимого буфера обмена, поскольку это средства обмена данными поддерживает большее число программ. Сделайте в программе пару элементов управления, перетаскивание которых в другие программы будет переносить текст или изображения.

Материалы: по ссылке на сайт имеются примеры. Для тестирования удобнее всего будет использовать Windows, где в качестве источника или приёмника переносимых данных использовать Microsoft Word.

Дополнительный вопрос: поскольку ваша задача не требует сборки библиотеки, вам необходимо продемонстрировать результат успешной сборки библиотеки mpfr.

4.18. lzo

Сайт: `http://www.oberhumer.com/opensource/lzo/`

Сборка: установлена в виртуальной машине, вручную — стандартным образом.

Задание: распакуйте или запакуйте указанный пользователем файл одним из алгоритмов lzo.

Материал: в библиотеке имеется множество алгоритмов с одинаковым интерфейсом, различающихся по производительности и сжатию. Возьмите алгоритм lzo1z_999, его описания даны в файле `lzo/lzo1z.h`: `lzo1z_999_compress` и `lzo1z_decompress`. Объём временных данных для работы алгоритма сжатия задаётся макросом `LZO1Z_999_MEM_COMPRESS`, для распаковки это 0. Полный пример можно найти в архиве дистрибутива под именем `examples/simple.c`.

4.19. libsodium

Сайт: `http://doc.libsodium.org/`

Сборка: с помощью менеджера пакетов: `emerge libsodium`, вручную — стандартным образом.

Задание: зашифруйте файл, указанный пользователем, или расшифруйте, убедившись, что он не был изменён злоумышленником после шифрования.

Материал:

- Чтобы преобразовать введённый пользователем текстовый пароль в ключ алгоритма шифрования, необходимо произвести его хеширование вместе с некоторыми дополнительными случайными данными («солью»), уникальными для каждого шифруемого файла и сохраняемыми вместе с ним. Пример этой процедуры приведён в примере 1 по адресу `http://doc.libsodium.org/password_hashing/README.html`.
- Работа с алгоритмом аутентифицированного шифрования показана по адресу `http://doc.libsodium.org/secret-key_cryptography/authenticated_encryption.html`. Дополнительные данные `nonce` также потребуется сохранить с зашифрованным файлом. Поскольку библиотека не поддерживает шифрование файлов по частям, считывать входной файл по блокам, как предлагается для общего случая, не нужно, достаточно корректно обработать ситуацию, когда памяти для обработки файла целиком не хватит.
- При дешифровке необходимо считать из зашифрованного файла соль, восстановить ключ шифрования по паролю, введённому пользователем, считать из зашифрованного файла `nonce`, а дальше расшифровать сами данные.

4.20. ntl

Сайт: `http://www.shoup.net/ntl/`

Сборка: с помощью менеджера пакетов:

```
echo ”dev-libs/gf2x @clang” >> /etc/portage/package.vars
echo ”dev-libs/ntl @clang” >> /etc/portage/package.vars
emerge ntl
```

Вручную собирать слишком сложно. Вместо этого продемонстрируйте сборку библиотеки libsodium.

Задание: вычислите характеристический многочлен матрицы из целых чисел, введённой пользователем в таблицу изменяемого размера.

Материал: вам потребуются классы для работы с многочленами (`http://www.shoup.net/ntl/doc/ZZX.cpp.html`) и матрицами (`http://www.shoup.net/ntl/doc/mat_ZZ.cpp.html`) из целых чисел. Матрица из целых чисел является специализацией шаблона класса матриц: `http://www.shoup.net/ntl/doc/matrix.cpp.html`. Вычисление характеристического многочлена есть единственная функция из отдельного заголовочного файла: `http://www.shoup.net/ntl/doc/mat_poly_ZZ.cpp.html`. Для ввода элементов матрицы используйте виджет `QTableWidget`.

4.21. libsndfile

Сайт: `http://www.mega-nerd.com/libsndfile/`

Сборка: с помощью менеджера пакетов: `emerge libsndfile`, вручную — слишком сложно. Вместо этого продемонстрируйте успешную сборку библиотеки libstatgrab.

Задание: выведите амплитудный график звуковых данных из указанного пользователем файла формата WAV.

Материал: библиотека позволяет получить доступ к звуковым файлам различных форматов (mp3 отсутствует по лицензионным соображениям) в распакованном виде. В таком виде данные представлены в формате Pulse Code Modulation: с некоторой частотой значение входного сигнала на каждом канале преобразовывается в число в некотором диапазоне значений и сохраняется. Например, PCM-формат «44100 герц, 16-бит, стерео» означает, что 44100 раз в секунду значение сигнала на двух каналах замеряется и кодируется числами от 0 до $2^{16} - 1$. Одно такое число библиотека называет элементом (item), а набор элементов для всех каналов в одну единицу времени — фреймом (frame). Для корректной обработки не-ASCII имён файлов, передайте дескриптор открытого классом `QFile` файла библиотечной функции `sf_open_fd`. В структуру `SF_INFO`, переданную ей, будет занесено в том числе число фреймов в файле и число каналов. Для чтения фреймов используйте функций `sf_readf_double`, она преобразует данные к диапазону $[-1; 1]$, независимо от формата файла. Достаточно учитывать данные только одного канала, если их в файле несколько. Выберите некоторую ширину графика для вывода на экран и усредните данные фреймов на выбранное число частей.

4.22. fftw

Сайт: `http://www.fftw.org`

Сборка: с помощью менеджера пакетов: `emerge fftw`, вручную — стандартным образом.

Задание: выведите результат применения дискретного преобразования Хартли к выбранному пользователем изображению.

Материал: для загрузки изображения используйте класс `QImage`. Для упрощения дальнейшей обработки приведите его к формату `QImage::Format_RGB8888` — по 8 бит на цветовую составляющую (красную/зелёную/синюю) и ещё один байт — с помощью функции `QImage::convertToFormat`. Теперь доступ к данным отдельных строк пикселей можно получить с помощью функции `QImage::scanLine`, приведите возвращаемое ей значение к `const QRgb*`. Бибилотека fftw обрабатывает данные в формате double, приведите значения типа `QRgb` для пикселей изображения к одной компоненте яркости функцией `qGray` и поделите на 255., чтобы получить значения в диапазоне $[0; 1]$, их сохраните в двумерный массив с шаговым доступом.

Для использования библиотеки сначала необходимо «запланировать» требуемое преобразование. В нашем случае понадобится функция `fftw_plan fftw_plan_r2r_2d` (`http://www.fftw.org/fftw3_doc/Real_002dto_002dReal-Tran html`). Поскольку необходимы преобразования в `double` из `QRgb`, этот же массив можно использовать и для выходных данных, т.е. совершить преобразование на месте. Созданный план можно выполнить функцией `fftw_execute` (`http://www.fftw.org/fftw3_doc/Using-Plans.html`). Просканируйте полученные данные и найдите их диапазон значений. Приведите эти данные назад к интервалу $[0; 255]$, и с помощью функции `qRgb` сделайте из них значения типа `qRgb`, передав одно и то же значение во все 3 параметра, вы получите соответствующие оттенки серого. Эти значения запишите назад в другое изображение, созданное с тем же размером, что и исходное. Отобразите оба изображения в виджетах `QLabel`, задав их через свойство `pixmap`.

4.23. qrencode

Сайт: `http://fukuchi.org/works/qrencode/`

Сборка: с помощью менеджера пакетов: `emerge qrencode`, вручную — стандартным образом.

Задание: отобразите на экране QR-код для введённой пользователем текстовой строки.

Материал: кодирование осуществляется функцией `QRcode_encodestring`. Вызывайте её с параметром `hint`, равным `QR_MODE_8` для прямого кодирования 8-битных значений. Для кодирования текста в UTF-8 используйте метод `QString::toUtf8`.

Результатом вызова функции является указатель на структуру типа `QRcode`. Она содержит указатель на квадратный массив символов указанной в нём ширины, в котором младший бит каждого значения определяет цвет соответствующего элемента кода. Для вывода их на экран создайте объект `QImage` нужного размера, заполните его требуемыми данными, и выведите на экран в виджете `QLabel`, задав изображение через свойство `pixmap`.

4.24. QtNetwork: DNS

Сайт: `http://qt-project.org/doc/qt-5/qdnslookup.html`

Сборка: не требуется.

Задание отобразите всю IPv4 и IPv6 адреса заданного пользователем хоста, а также имеющуюся информацию о серверах, обслуживающих его записи DNS и почту.

Материал: получите и отобразите в виде текста или таблицы всю информацию, полученную для запросов типа A, AAAA, NS и MX. Пример использования класса приведён по данной выше ссылке.

Дополнительный вопрос: поскольку ваша задача не требует библиотеки помимо Qt, вам необходимо продемонстрировать результат успешной сборки библиотеки qrencode.

4.25. lz4

Сайт: `https://code.google.com/p/lz4/`

Сборка: с помощью менеджера пакетов: `emerge lz4`, вручную — стандартным образом. Вместо этого продемонстрируйте успешную сборку библиотеки fftw.

Задание: Распакуйте или запакуйте указанный пользователем файл с помошью библиотеки lz4.

Материал: изучите документацию в заголовочном файле `lz4.h` для функций `LZ4_compress` и `LZ4_decompress_safe`. Они осуществляют сжатие/распаковку отдельных блоков, на которые вам понадобится разбить входной файл.

Сборка библиотеки для Windows:

- Скачайте и распакуйте исходные тексты библиотеки по ссылке с заглавной страницы сайта.
- Запустите командную строку и добавьте путь к компилятору, установленному вместе с Qt Creator, в путь поиска программ. Это подкаталог `Tools\mingwXXX_32\bin` каталога его установки, содержащий сам компилятор `gcc.exe`. Для добавления к пути поиска используется команда `path добавляемый_каталог;%path%`.
- Перейдите в каталог распакованной библиотеки и выполните команды:

```
gcc -c -std=c99 lz4.c lz4hc.c
ar rcs liblz4.a lz4.o lz4hc.o
```

- Эти команды создают статическую библиотеку с именем `lz4`. Укажите её для использованию компоновщиком, а также добавьте каталог с распакованной библиотекой к путям поиска заголовочных файлов и библиотек, как описано в общих сведениях выше.

4.26. QtNetwork: защищённые соединения

Сайт: `http://qt-project.org/doc/qt-5/ssl.html`

Сборка: не требуется.

Задание: установите защищённое соединение по протоколу TLS с указанной пользователем точкой и выведите информацию о сертификатах удалённой стороны.

Материал: для установления защищённого соединения используется метод `QSslSocket::connectToHostEncrypted`. После получения сигнала `encrypted()` можно получить доступ к цепочке сертификатов с помощью вызова `QSslSocket::peerCertificateChain`. Этот список содержит объекты типа `QSslCertificate`, из них можно получить все требуемые свойства. Используйте виджет `QTableWidget` для отображения имени субъекта, дат действия сертификата, алгоритма и длины ключа.

Дополнительный вопрос: поскольку ваша задача не требует библиотеки помимо Qt, вам необходимо продемонстрировать результат успешной сборки библиотеки lz4.

4.27. flint

Сайт: `http:/flintlib.org`

Сборка: с помощью менеджера пакетов: `emerge flint`, вручную — стандартным образом.

Задание: разложите натуральное число произвольной длины на простые множители и выведите результат в форме произведения, например $600 = 2^3 \cdot 3 \cdot 5^2$.

Материал: вам потребуются функции семейства `fmpq` (раздел 18) и `fmpz_factor` (раздел 20) документации: `http://flintlib.org/flint-2.4.4.pdf`. Формат результата — указатель на структуру `fmpz_factor_t`, содержимое структуры — `num` множителей, где `p` — множители, а `exp` — показатели степени при них (см. `fmpz_factor.h`).

Для вывода результата вам потребуется виджет отображения форматированного текста для вывода верхних индексов степеней — `QTextBrowser`. Для добавления текста используйте метод `QTextEdit::appendHtml`, заключая показатели степени в теги `<sup>`.

Чтобы собрать библиотеку для Windows, потребуется перед этим установить две её зависимости: `gmp` и `mpfr`. Все три библиотеки необходимо собирать в среде MSYS, предоставляющей UNIX-подобные средства для этапа конфигурации.

- Скачайте и распакуйте исходные тексты всех трёх библиотек, ссылки на сайты имеются в этой инструкции.
- Распакуёте архив с системой MSYS: `http://sourceforge.net/projects/mingwbuilds/files/external-binary-pacl msys%2B7za%2Bwget%2Bsvn%2Bgit%2Bmercurial%2Bcvs-rev13.7z/download`.
- В файле `etc\fstab` укажите путь к MinGW от Qt Creator с прямыми слешами, отображённый в `/mingw`. Для случая с Qt Creator, установленным в каталог `C:\Qt` и версией компилятора 4.8.2:

  ```
  C:/Qt/Tools/mingw482_32 /mingw
  ```

- Запустите `msys.bat`, чтобы получить доступ к командной строке системы. Содержимое дисков ОС видно в путях, начинающихся с корня и одной строчной буквы диска, например, вместо `C:\a.txt` следует писать `/c/a.txt`. В таком стиле необходимо указывать все пути во всех командах.
- Установите скачанные пакеты по инструкции в разделе «Стандартная сборка пакетов», в порядке `gmp`, `mpfr`, `flint`. В качестве команды конфигурации используйте

  ```
  CFLAGS=-I/local/include LDFLAGS=-L/local/lib ./configure --enable-static --disable-shared
  ```

  чтобы собрать статические версии библиотек (проще в использовании для Windows), установить их в префикс MSYS по умолчанию и добавить его к путям поиска зависимых библиотек.
- Подкаталоги `local\include` и `local\lib` каталога установки системы MSYS теперь содержат заголовочные файлы и собранные библиотеки соответственно, их необходимо указать в файле проекта, как указано в разделе «Общие рекомендации». Также необходимо указать все три библиотеки для компоновки в порядке, обратном порядку их сборки.

4.28. QtNetwork: сетевые интерфейсы

Сайт: `http://qt-project.org/doc/qt-5/qnetworkinterface.html`

Сборка: не требуется.

Задача: выведите в табличном виде информацию о сетевых интерфейсах.

Материал: информацию о сетевых интерфейсах можно получить с помощью вызова `QNetworkInterface::allInterfaces`. Для отображения информации используйте виджет `QTableView`. Для предоставлению ему информации реализуйте класс, производный от `QAbstractTableModel`, отразите в ней имена интерфейсов, их аппаратные адреса и первый назначенный IP-адрес. Дополнительная информация о поддержке разделения модель-вид в библиотеке Qt: `http://qt-project.org/doc/qt-5/model-view-programming.html`.

Дополнительный вопрос: поскольку ваша задача не требует библиотеки помимо Qt, вам необходимо продемонстрировать результат успешной сборки библиотеки flint.

4.29. hunspell

Сайт: `http://hunspell.sourceforge.net/`

Сборка: с помощью менеджера пакетов:

```
echo ”app-text/hunspell @clang” >> /etc/portage/package.vars
echo ”app-text/hunspell linguas_en linguas_ru” >> /etc/portage/package.use/local
emerge hunspell
```

Сборка вручную — стандартным образом.

Задание: проверьте орфографию введённого пользователем текста и предложите варианты исправления для ошибок.

Материал: документация содержится в самих заголовочных файлах `hunspell.h/hunspell.hxx` или по адресу `http://sourceforge.net/projects/hunspell/files/Hunspell/Documentation/hunspell3.pdf/download` (несколько запутанная, для C++), вы можете использовать интерфейс как на C, так и на C++. При инициализации библиотеки необходимо указать пути к файлам аффиксов и словарю.

В результате приведённых команд для менеджера пакетов будут установлены словари английского и русского языков в каталог `/usr/share/myspell`. Файлы для русского языка устанавливаются в неправильной кодировке, исправить это можно следующим образом:

```
cd /usr/share/myspell
for f in *ru_RU* ; do iconv -f koi8-r $f > ${f}.new ; mv ${f}.new ${f} ; sed -i 's/KOI8-R/UTF-8/' $f ; done
```

Функция `Hunspell_spell` или метод `spell` позволяют проверить отдельное слово на корректность, а `Hunspell_suggest` или метод `suggest` позволяют получить список исправлений для слова с ошибкой, который после использования надо освободить функцией `Hunspell_free_list` или методом `free_list`.

Всё взаимодействие с библиотекой осуществляется в кодировке utf-8, для работы с ней используйте методы `QString::toUtf8` и `QString::fromUtf8`. При разбиении текста на слова можно воспользоваться функцией `QChar::isLetter`, которая работает для любых языков, поддерживаемых Unicode.

4.30. libnova

Сайт: `http://libnova.sourceforge.net/`

Сборка: с помощью менеджера пакетов: `emerge libnova`, вручную:

```
libtoolize --install --copy --force --automake
aclocal
autoconf
autoheader
automake --add-missing --copy --force-missing
CC=clang ./configure --prefix=/home/user/prefix --enable-shared --disable-static
```

Задание: по указанным пользователем времени, дате и географическому положению, отобразите положение Луны и планет солнечной системе на небе, видимом наблюдателю.

Материал: вначале потребуется перевести время к формату используемому библиотекой. Для ввода информации пользователем можно использовать виджет `QDateTimeEdit`, результат ввода имеет тип `QDateTime` и доступен в свойстве `dateTime`. Его можно преобразовать к типу `time_t` вызовом `QDateTime::toTime_t`, а это значение, в свою очередь, к типу `double`, используемому библиотекой для хранения Юлианского дня, функцией `ln_get_julian_from_timet` (`http://libnova.sourceforge.net/group__calendar.html#gb382e9eaf1e6a10d346a31d598`

Для каждого небесного тела библиотека имеет свой набор функций: `http://libnova.sourceforge.net/modules.html`. Для каждого из них есть функций вычисления экваториальных координат по заданному Юлианскому дню, например, для Луны — `ln_get_linar_equ_coords` (`http://libnova.sourceforge.net/group__lunar.html#gc9ddb9e6b5b`
Полученное значение типа `ln_equ_posn` можно преобразовать к горизонтальным координатам типа `ln_hrz_psn`, исходя из положения наблюдателя типа `ln_lnlat_posn` и времени функцией `ln_get_hrz_from_equ`. Полученный азимут и склонения можно привести к координатам точки на плоскости: отрицательные склонения находятся за горизонтом и не должны выводится. Положительные находятся в интервале $[0; 90]$ и должны быть отображены на всю высоту картинки, а азимуты лежат в $[0; 360)$ и должны быть отображены на всю ширину картинки. В вычисленных местах нарисуйте небольшие окружности соответствующих цветов или более сложные изображения.

Поскольку необходимые функции вычисления координат имеют одинаковый тип для всех небесных тел, можно поместить их в массив указателей на функции для записи всего процесса вычислений одним циклом.