Лекция 1. Знакомимся с языком C++

В этой лекции мы напишем первую программу, познакомится с основными этапами перевода программы с языка C++ в машинный код и познакомимся со средами программирования в Linux.

1.1. Первая программа на C++

Знакомство с языком C++ начнем с решения простейшей задачи.

ЗАДАЧА 1.1. Заданы длины двух катетов прямоугольного треугольника a, b. Вычислить длину гипотенузы c и величины двух его углов ? и ? .Значения a, b, c ввести с клавиатуры.

Перед написанием программы давайте вспомним основные формулы, которые нам понадобятся. Гипотенуза c вычисляется по формуле $c=\sqrt{a^2+b^2}$, углы треугольника α и β рассчитываются следующим образом: $\alpha=arctg\left(\frac{a}{b}\right)$, $\beta=\frac{\pi}{2}-\alpha$. Решение задачи можно разбить на следующие этапы:

1. Определение значений a, b (ввод величин a, b и c клавиатуры в память компьютера).
2. Расчет значений c, α и β по приведенным выше формулам.
3. Вывод значений c, α и β на экран дисплея.

Ниже приведен текст программы. Сразу заметим, что в тексте могут встречаться строки, начинающие с двух наклонных (//), являющиеся комментариями. *Комментарии* не являются обязательными элементами программы и ничего не сообщают компьютеру, они поясняют человеку, читающему текст программы, назначение отдельных элементов программы. В книге комментарии будут широко использоваться для пояснения отдельных участков программы.

```
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
double a,b,c,alf,bet,s;
cout<<"A=";
cin>>a;
cout<<"B=";
cin>>b;
s=a*b/2;
c=pow(a*a+b*b,0.5);
alf=atan(a/b);
bet=3.14159/2-alf;
cout<<"\n A="<<a<<"\t B="<<b<<"\t C="<<c<<"\n";
cout<<"\nS="<<s<<"\t alf="<<alf*180/3.14159;
cout<<"\t bet="<<bet*180/3.14159<<endl;
}
```

1.2. Неформальное введение в C++

Давайте построчно подробно рассмотрим текст программы и познакомимся со структурой программы на C++ и с некоторыми операторами языка.

Строки 1-2. Указывают компилятору (а точнее, препроцессору), что надо использовать функции из библиотек iostream и math.h.. Библиотека iostream нужна для организации ввода с помощью cin и вывода с помощью cout[1]. Для использования математических функций

1 Обратите внимание на особенность синтаксиса при подключении библиотеки iostream.

возведения в степень pow и вычисление арктангенса atan необходима библиотека математических функций math.h. В программе на языке С++ должны быть подключены все используемые библиотеки.

Строка 3. Эта строка обозначает, что при вводе и выводе с помощью cin и cout будут использоваться стандартные устройства (клавиатура и экран), если эту строку не указывать, то каждый раз при вводе вместо cin надо будет писать std::cin, а вместо cout std::cout.

Строка 4. Заголовок главной функции.

Строка 5. Любая функция начинается с символа {.

Строка 6. Описание вещественных переменных a, b, c, alf, bet, s. *Имя переменной (идентификатора)* состоит из латинских букв, цифр и символа подчеркивания. Имя не может начинаться с цифры. В языке С++ большие и малые буквы различимы, имена PR_1, pr_1, Pr_1 и pR_1 - разные.

Строка 7. Вывод строки символов A= с помощью cout.

Строка 8. Ввод вещественного числа a с помощью cin

Строка 9. Вывод строки символов B= с помощью cout.

Строка 10. Ввод вещественного числа b с помощью cin.

Строка 11. Оператор присваивания для вычисления площади треугольника по формуле s=ab/2. В операторе присваивания могут использоваться знаки операций: `+, -, *, /`.

Строка 12. Оператор присваивания для вычисления гипотенузы с использованием теоремы Пифагора. Функция pow(x,y) используется в С++ для вычисления x^y.

Строки 13-14. Операторы присваивания для вычисления углов α и β по формулам $\alpha = arctg\left(\frac{a}{b}\right)$, $\beta = \frac{\pi}{2} - \alpha$.

Строки 15-17. Функции вывода результатов на экран.

Следует учитывать, что в тригонометрических функциях в С++ углы вычисляются в радианах. Для получения значений углов в градусах значения α и β умножаются на $\frac{180}{\pi}$.

Мы рассмотрели простейшую программу на языке С++, состоящую из операторов ввода данных, операторов присваивания (в которых происходит расчет по формулам) и операторов вывода.

Любая программа на языке С++ представляет собой одну или несколько функций. В любой программе обязательно должна быть одна функция main(). С этой функции начинается выполнение программы. Правилом хорошего тона в программировании является разбиение задачи на подзадачи, и в главной функции чаще всего должны быть операторы вызова других функций. Общую структуру любой программы на языке С++ можно записать следующим образом.

```
Директивы препроцессора
Объявление глобальных переменных
Тип_ результата main(Список_переменных)
{
Операторы
}
Тип_результата f1(Список_переменных)
{
Операторы
}
```

```
Тип_результата f2(Список_переменных)
{
Операторы
}
...
Тип_результата fn(Список_переменных)
{
Операторы
}
```

Здесь `Тип_результата` - тип возвращаемого функцией значения.

В простейшем случае программа на языке Си состоит из одной функции main, в этом случае структура программы будет такой.

```
int main()
{
Операторы
}
```

Введенная в компьютер программа на языке С++ должна быть переведена в двоичный машинный код (должен быть сформирован файл с расширением **.exe**). Для этого существуют специальные программы, называемые трансляторами. Все *трансляторы* делятся на два класса:

- *интерпретаторы* - трансляторы, которые переводят каждый оператор программы в машинный код, и по мере перевода операторы выполняются процессором;
- *компиляторы* переводят всю программу целиком, и если перевод всей программы прошел без ошибок, то полученный двоичный код можно запускать на выполнение.

Процесс перевода программы в машинный код называется *трансляцией*. Если в качестве транслятора выступает компилятор, то используют термин *компиляция* программы. При переводе программы с языка С++ в машинный код используются именно компиляторы, и поэтому применительно к языку С++ термины «компилятор» и «транслятор» эквивалентны.

Рассмотрим основные этапы обработки компилятором программы на языке С++ и формирования машинного кода.

1. Сначала программа обрабатывается `препроцессором`[2], который обрабатывает директивы препроцессора, в нашем случае это директивы включения заголовочных файлов (файлов с расширением **.h**) - текстовых файлов, в которых содержится описание используемых библиотек. В результате формируется полный текст программы, который поступает на вход компилятора.
2. Компилятор разбирает текст программ на составляющие элементы, проверяет синтаксические ошибки и в случае их отсутствия формирует объектный код (файл с расширением **o**). Получаемый на этом этапе двоичный код не включает в себя двоичные коды библиотечных функций и функций пользователя.
3. *Компоновщик* подключает к объектному коду программы объектные модули библиотек и других файлов (если программа состоит из нескольких файлов) и генерирует исполняемый код программы, который уже можно запускать на выполнение. Этот этап называется компоновкой или сборкой программы. После этого исполняемый файл можно запускать на выполнение.

После написания программы ее необходимо ввести в компьютер. Рассмотрим наиболее часто используемые в ОС Linux среды разработки программ.

---

2 Препроцессор это программа ,которая преобразовывает текст директив препроцессора в форму ,понятную компилятору. О данных на выходе препроцессора говорят, что они находятся в препроцессированной форме.

1.3. Средства разработки программ в ОС Linux

При разработке консольных программ на C++ в Ubuntu можно использовать следующие средства разработки.

1. Текстовый редактор совместно с компилятором gcc (g++).
2. Интегрированная среда разработки Anjuta.
3. Интегрированная среда разработки Kdevelop.

1.3.1. Использование компилятора командной строки для создания консольных приложений

Для полноценной компиляции консольных приложений необходимо установить пакеты g++, g++-4.1, gcc, gcc-4.1. Это можно сделать с помощью с помощью менеджера пакетов Synaptic (см. рис. 1.1).

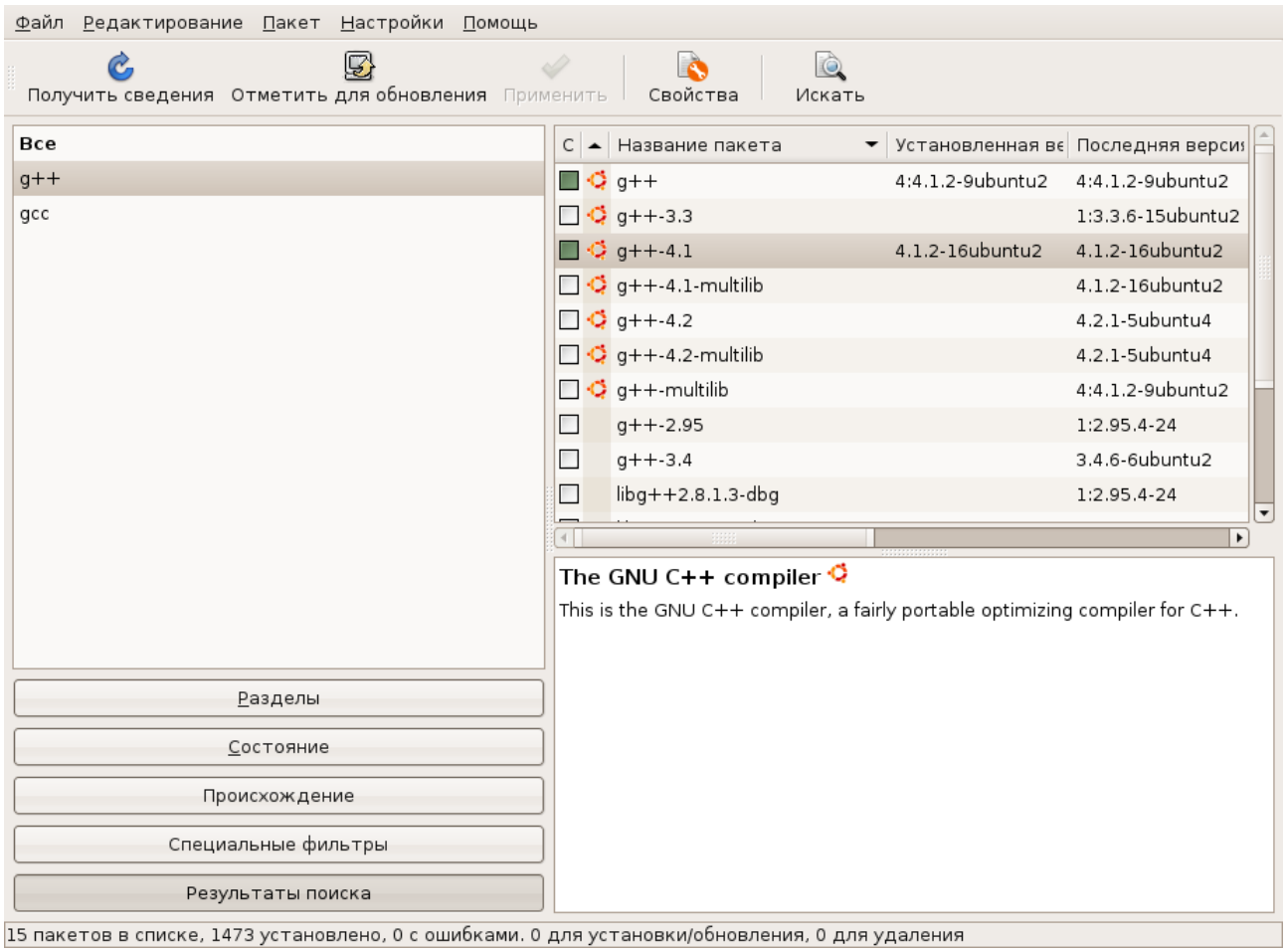

Рисунок 1.1: Окно менеджера пакетов Synaptic

После установки пакетов для создания программ можно использовать простейший текстовый редактор, входящий в состав Gnome и компилятор командной строки g++. К преимуществу стандартного текстового редактора можно отнести подсветку синтаксиса C++.

Рассмотрим опции компилятора командной строки, необходимые для компиляции и запуска простейших программ.

Для того, чтобы создать исполняемый файл из текста программы на C++, необходимо выполнить команду

g++ name.cpp

Здесь *name.cpp* имя файла с текстом программы .В результате будет создан исполняемый файл со стандартным именем *a.out*. Для того, чтобы создать исполняемый файл с другим именем, необходимо выполнить команду

g++ -o nameout name.cpp

Здесь *name.cpp* имя файла с текстом программы, *nameout* имя исполняемого файла .

При компиляции программ на C вместо компилятора g++ можно использовать компилятор gcc.
При использовании компиляторов gcc (g++) после компиляции программы автоматически происходит компоновка программы (запуск компоновщика *make*). Чтобы исключить автоматическую компоновку программы следует использовать опцию -c. В этом случае команда будет иметь вид

g++ -c name.cpp

Технология работы с компилятором g++ может быть такой: набираем текст программы в стандартном текстовом редакторе, потом в консоли запускаем компилятор, после исправления синтаксических ошибок, запускаем исполняемый файл, потом можно вносить изменения в текст программы. При такой технологии работы с компилятором, необходимо не забывать сохранять текст программы, иначе при запуске компилятора будет компилироваться старая версия текста программы.
Для разработки программ на различных языках программирования можно использовать текстовый редактор Geany, который есть в репозитарии Ubuntu. Разработка программ с использованием Geany более эффективна. Установка Geany также может быть осуществлена с менеджера пакетов Synaptic. Окно Geany представлено на рис. 1.2.

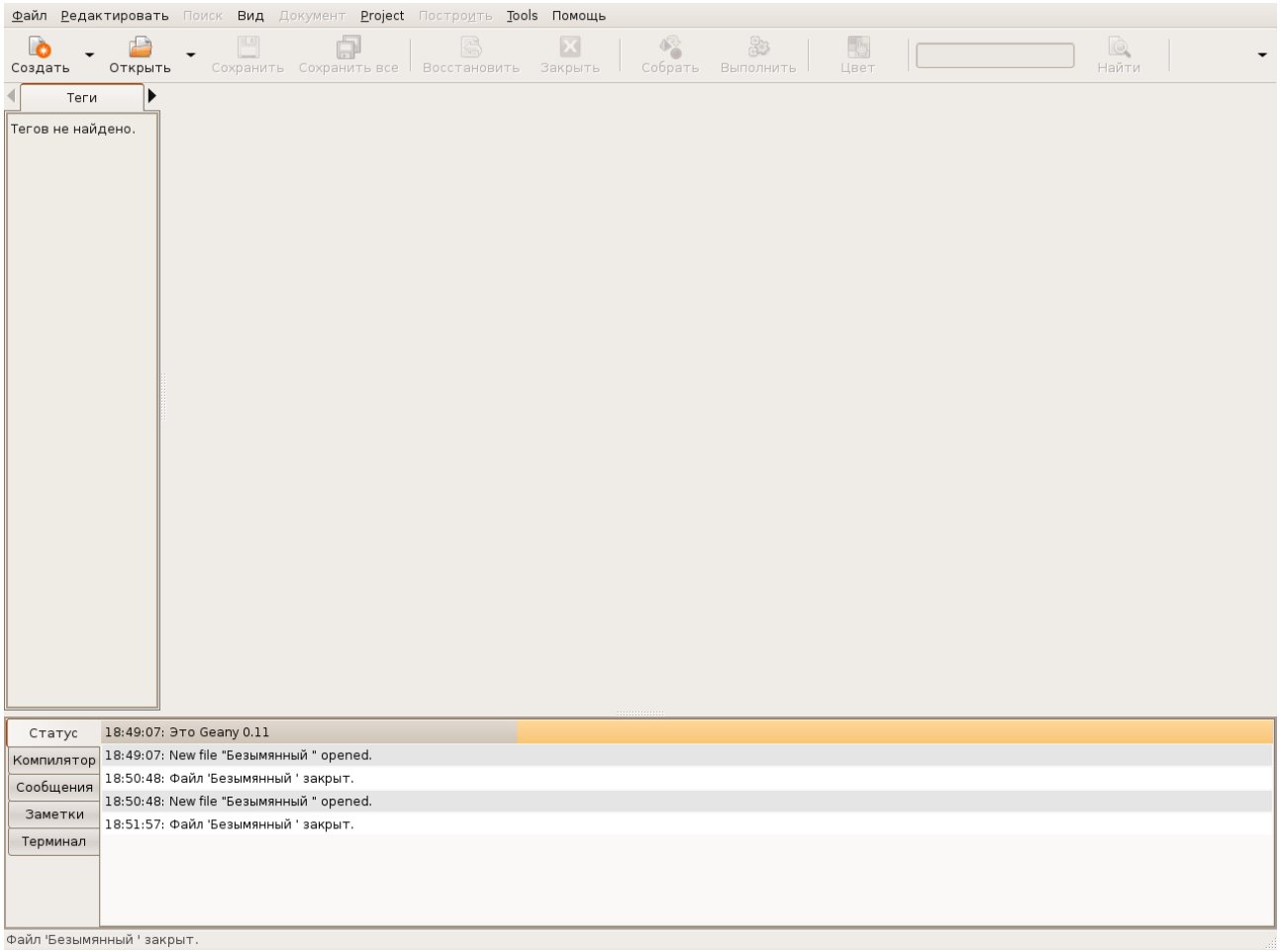

Рисунок 1.2: Окно Geany

Настройка
Последовательно рассмотрим основные этапы разработки программы с использованием Geany.

1. Необходимо создать шаблон приложения на C/C++ (или другом языке программирования) с помощью команды **Файл New (with Template) C ++ исходный код**. После чего появится окно с шаблоном исходного кода, после чего необходимо ввести текст программы и сохранить его (см. рис. 1.3).
2. Для компиляции и запуска программы на выполнение служит пункт меню **Построить**.

Для компиляции программы следует использовать команду **Построить Собрать (F8)**. В этом случае будет построен объектный код программы (файл с расширением **.o**).

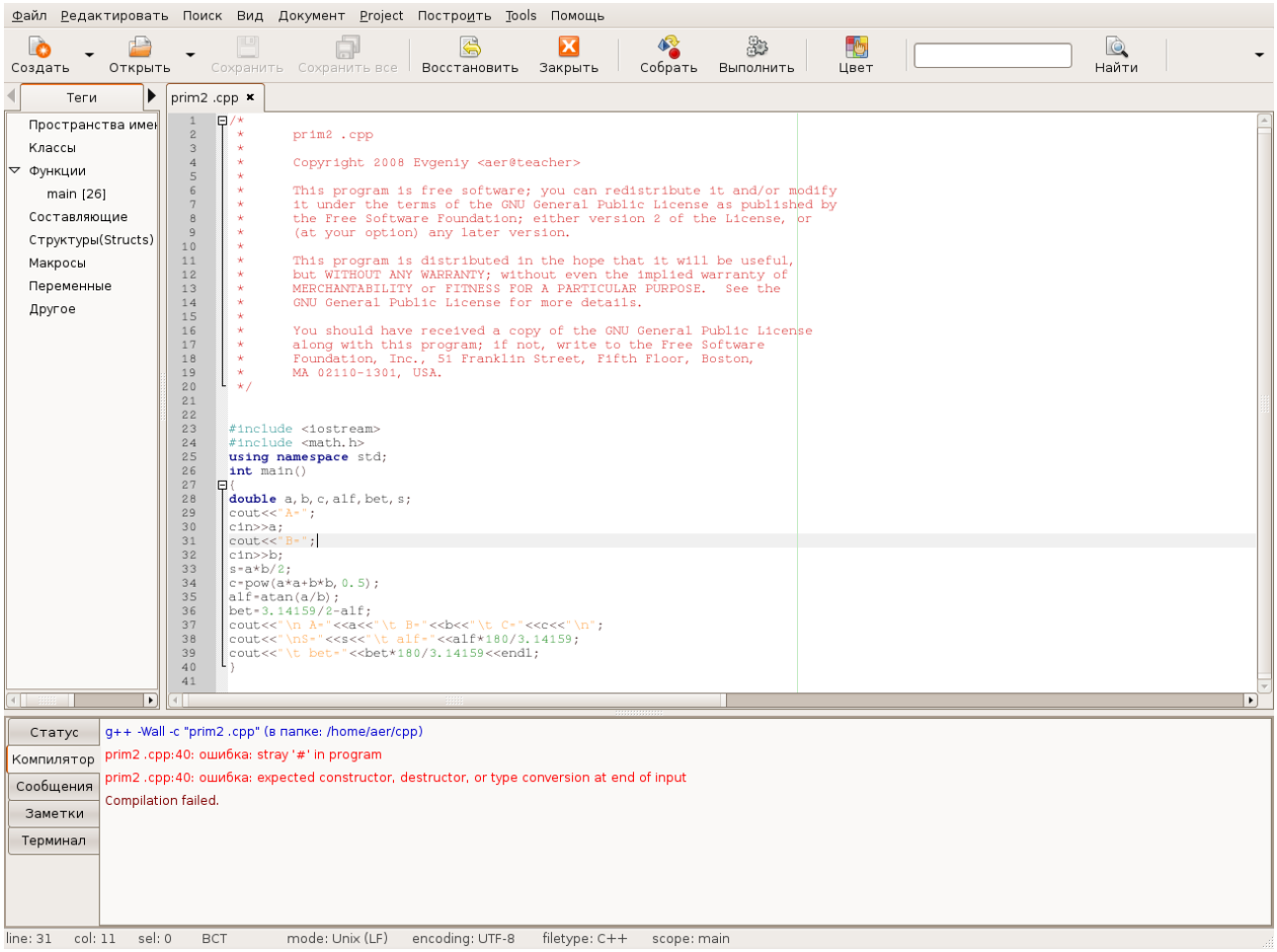

Рисунок 1.3: Окно Geany с текстом программы на C++

3. Для создания исполняемого кода программы служит команда **Построить Построить (F9)**.
4. Для запуска программы следует выполнить команду **Построить Выполнить (F5)**

Но следует помнить, что в редакторе Geany (хотя часто его называют и средой программирования) можно настроить какой командой вызывается компиляция, компоновка и запуск. Для это служит команда **Построить Установить включения и аргументы**. На мой взгляд значения по умолчанию следует поменять следующим образом (см. рис. 1.4).

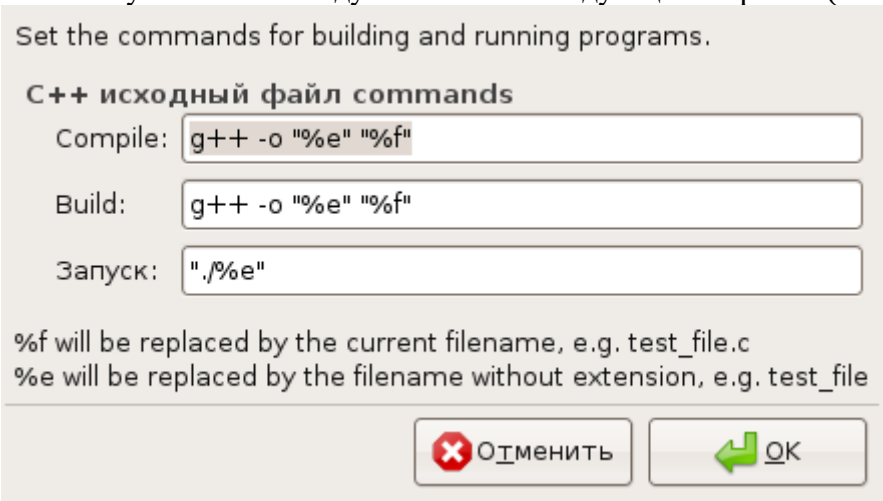

Рисунок 1.4: Настройка компиляции программ на C++в Geany

Здесь %f имя компилируемого файла , %e имя файла без расширения .
При запуске программы на выполнение (**Построить Выполнить**) появляется окно, подобное представленному на рис. 1.5.

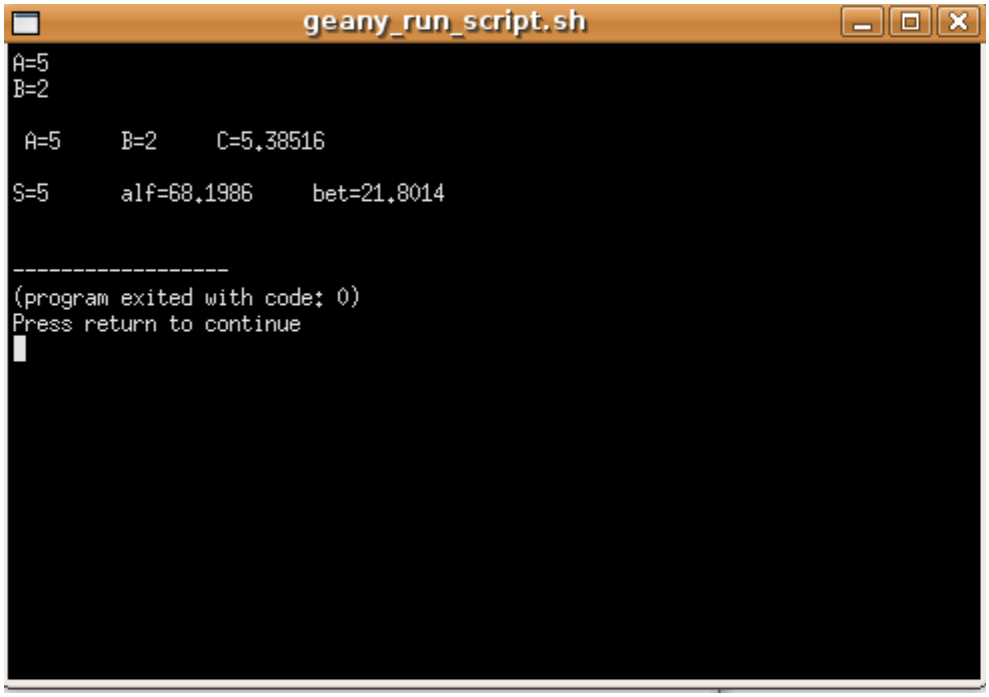

Рисунок 1.5: Запуск консольного приложения

1.3.2. Создание консольных приложений в среде Anjuta

В Ubuntu 7.10 установка приложения Anjuta из репозитария с помощью менеджера пакетов Synaptic проходит без проблем, после установки при первом запуске Anjuta сообщает, какие пакеты следует доставить, после установки которых все работает.
Проект в программе Anjuta создается с помощью команды Файл Новый Project. Основные этапы создания консольных приложений в среде Anjuta представлены на рис. 1.6-1.10.

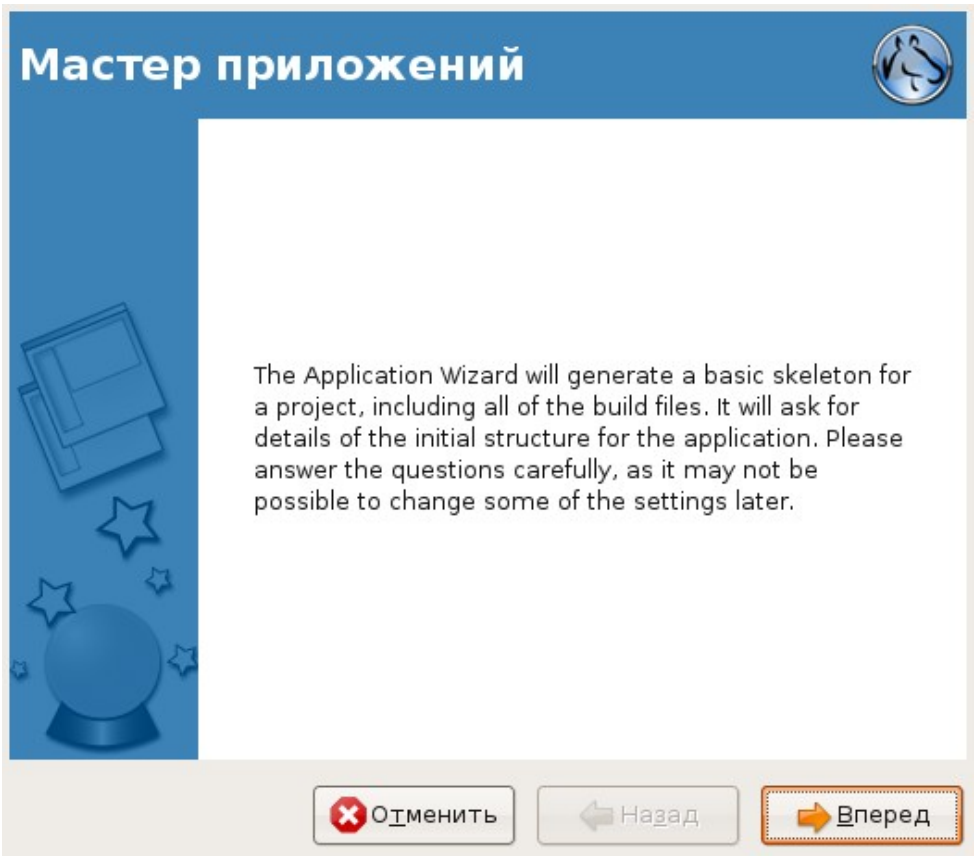

Рисунок 1.6: Первое окно мастера приложений

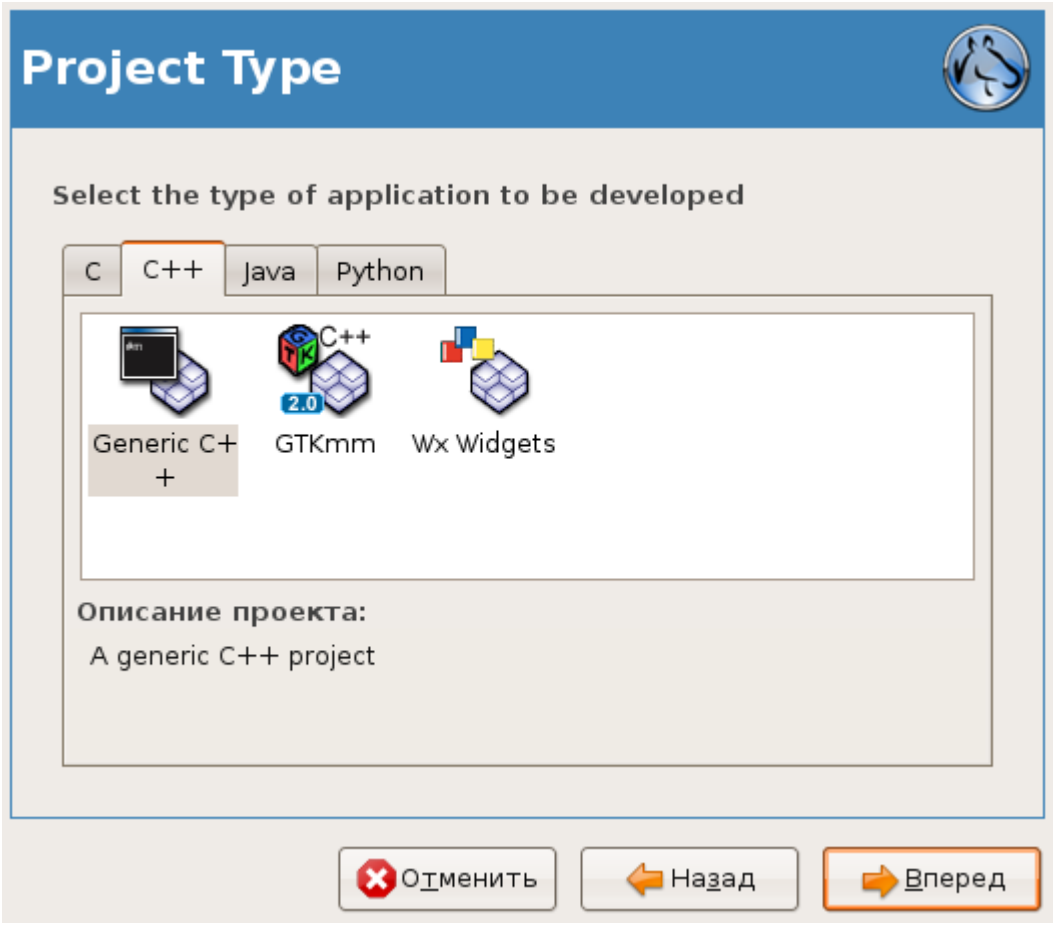

Рисунок 1.7: Выбор типа приложения

Рисунок 1.8: Основные параметры создаваемого проекта

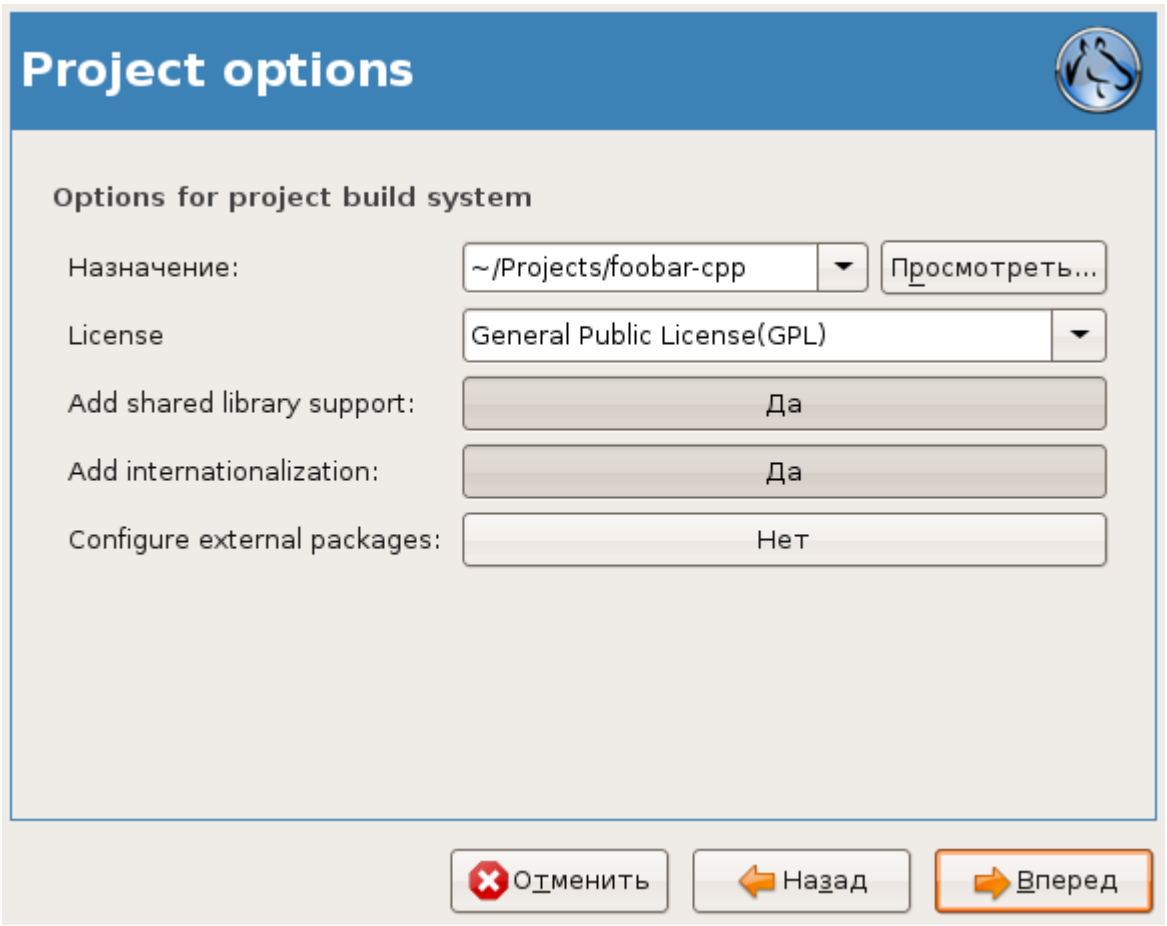

Рисунок 1.9: Параметры проекта (месторасположение, тип лицензии и т.д.)

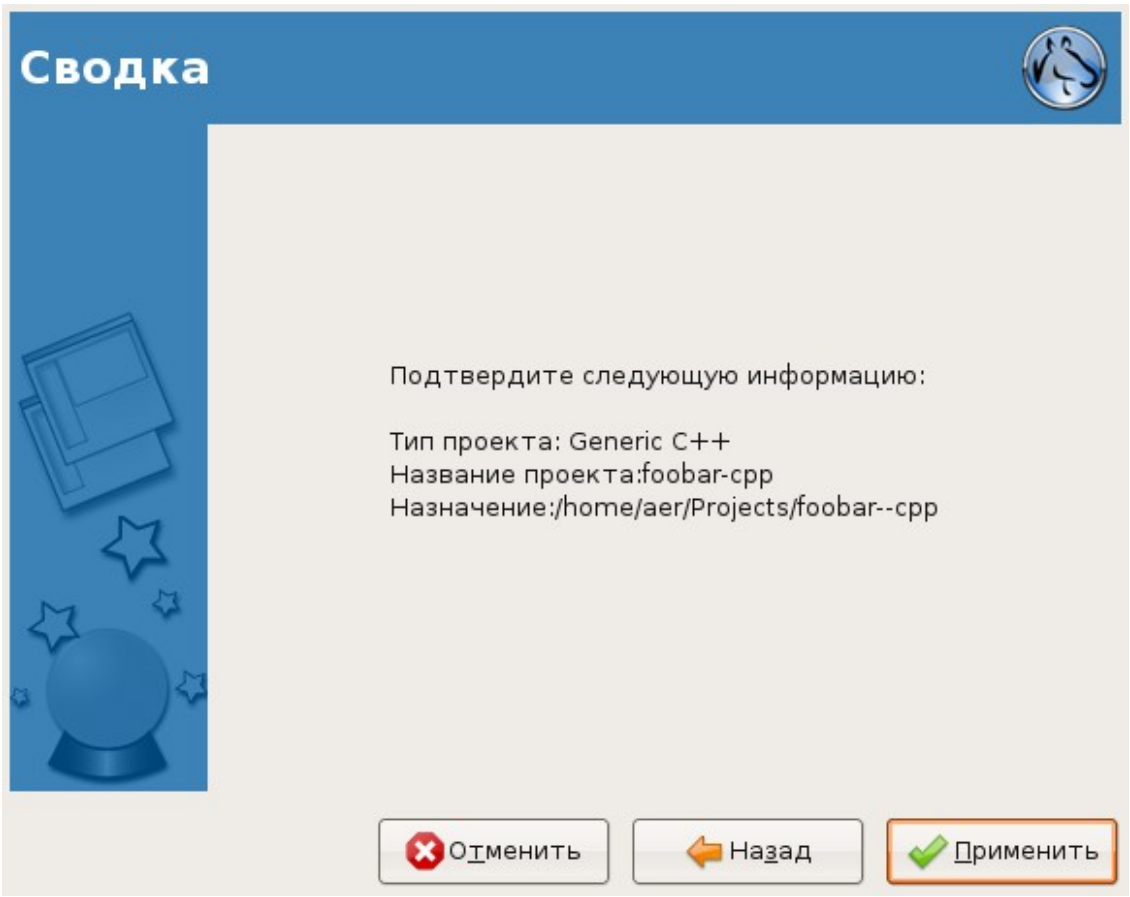

Рисунок 1.10: Подтверждение основных параметров проекта

На последнем этапе осуществляет выбор текстового редактора для работы с приложением (см. рис. 1.11). После чего пользователь видит окно проекта консольного приложения (см. рис. 1.12).

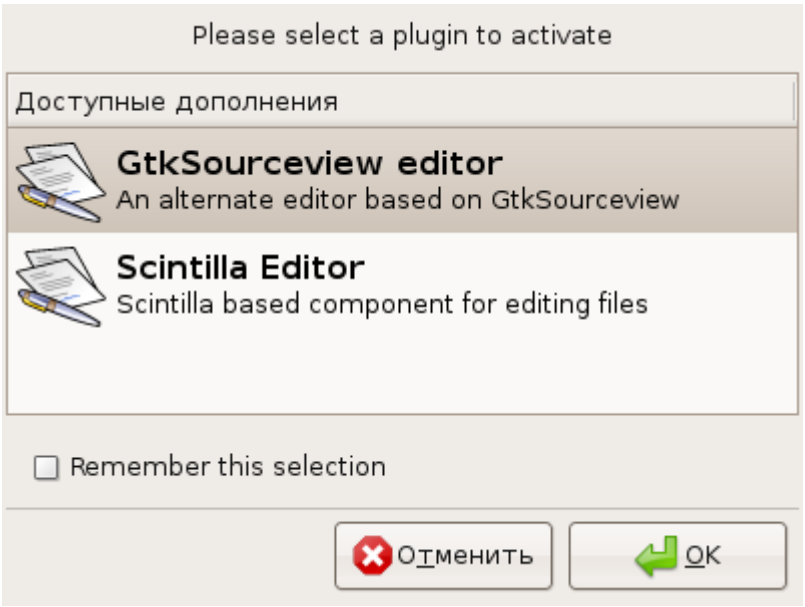

Рисунок 1.11: Выбор текстового редактора

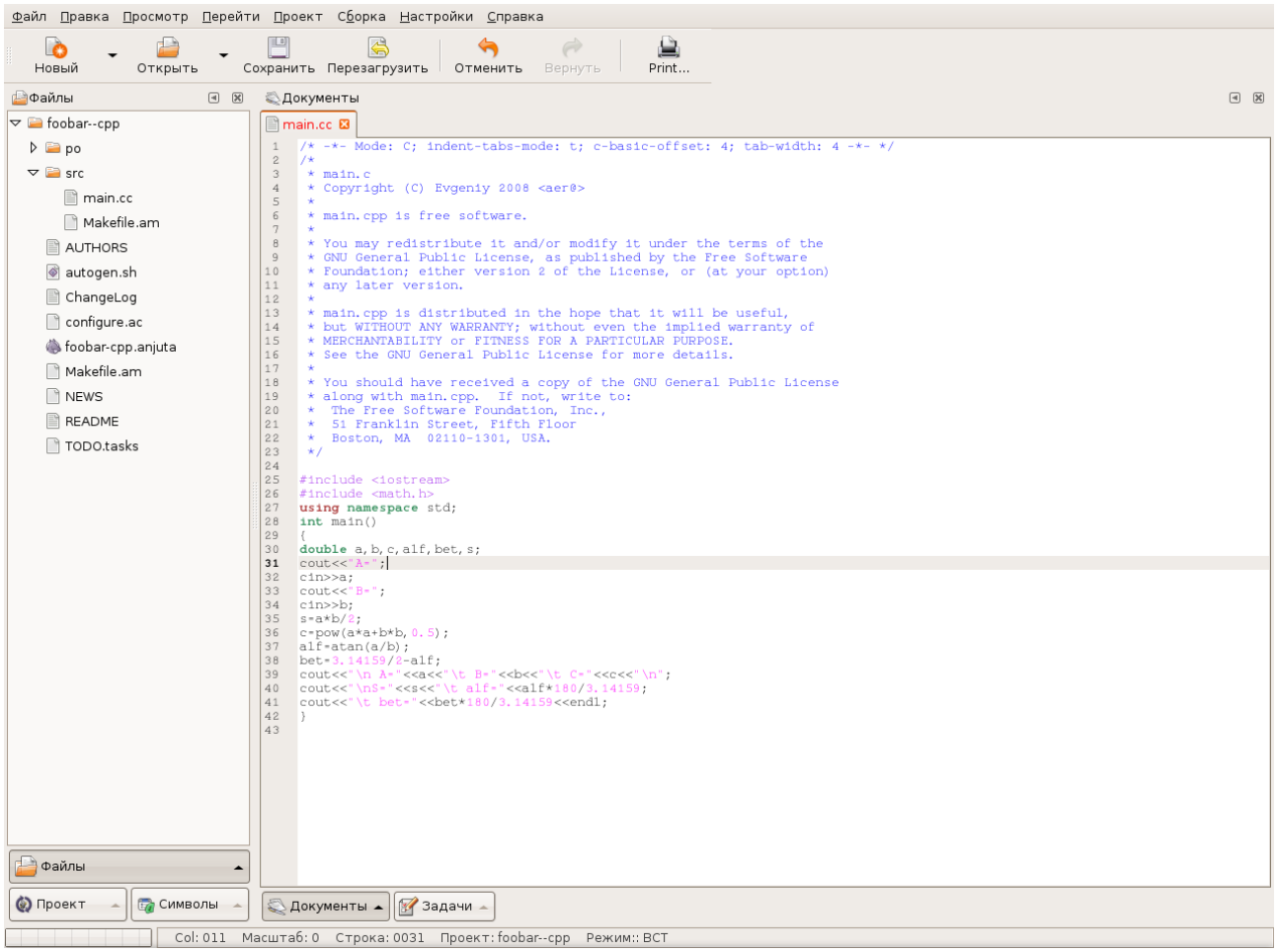

Рисунок 1.12: Окно проекта

Для компиляции и запуска программы на выполнение в среде Anjuta служит пункт меню **Сборка**. Для компиляции программы следует использовать команду **Сборка Компилировать (F9)**. Для сборки проекта служит команда **Сборка Сборка (F11)** Запуск программы осуществляется с помощью команды **Сборка Execute Program (F3)** После чего ниже окна проекта открывается окно терминала консольного приложения (см. рис. 1.13).

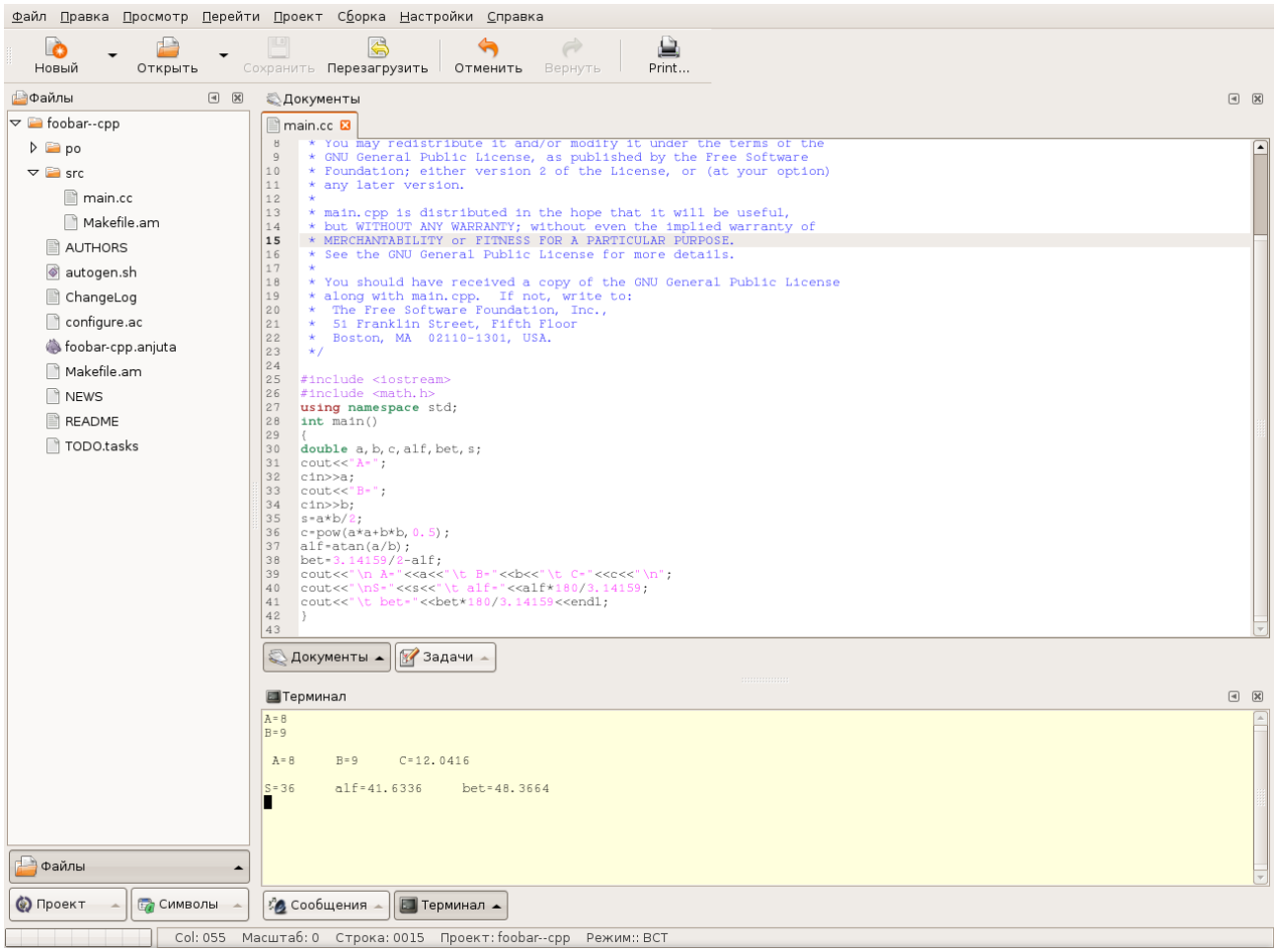

Рисунок 1.13: Результаты работы консольного приложения