

Плагин PascalABC.NET для интегрированной среды Visual Studio Code



Костикова Оксана

Южный Федеральный Университет

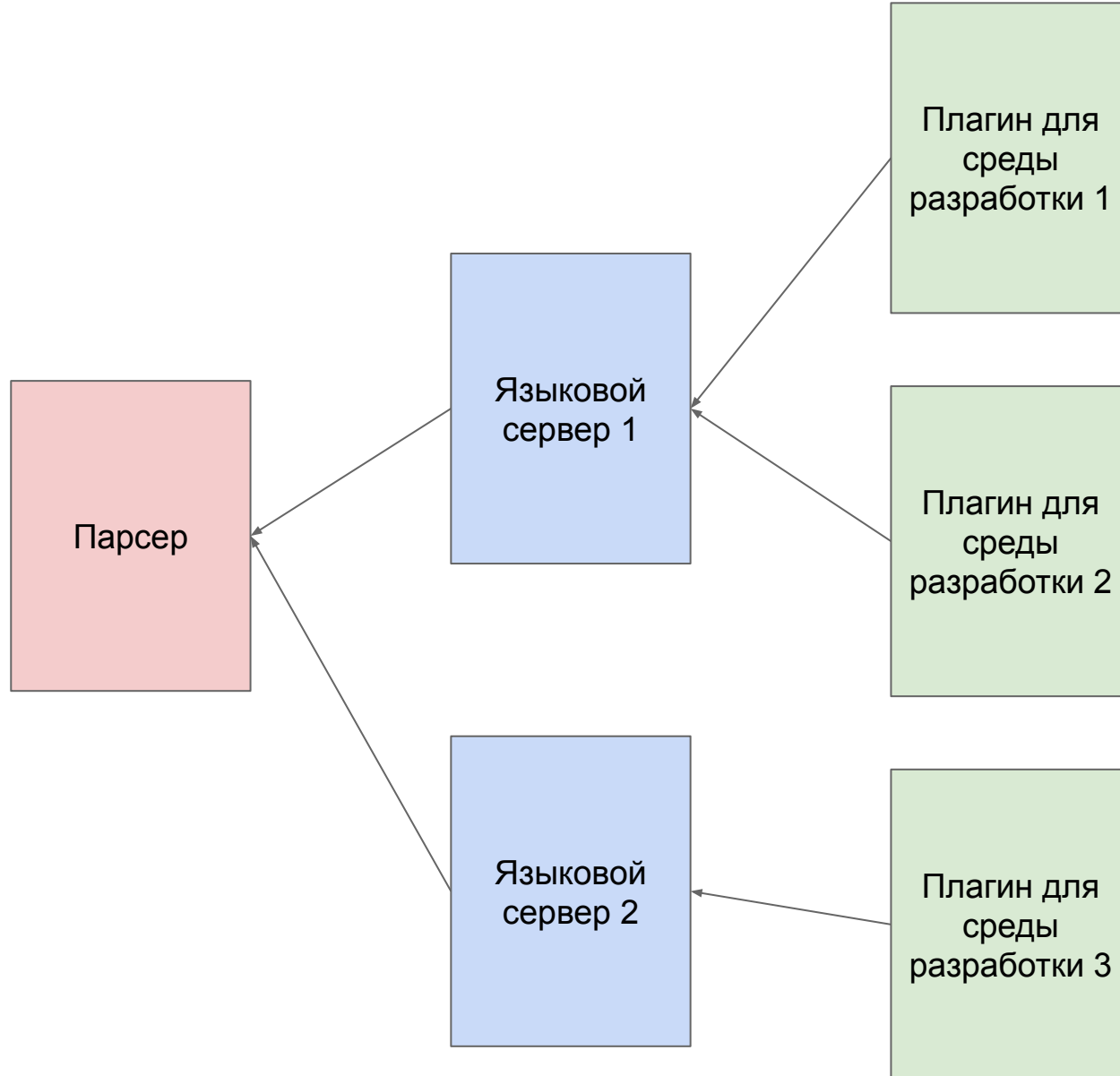
Институт математики, механики и компьютерных наук им И. И. Воровича

kostikova@sfedu.ru

Зачем это нужно?

- Предоставить пользователям Linux удобный инструмент для разработки на PascalABC.NET
- Создать основу для более активного развития инструментов экосистемы PascalABC.NET
- Расширить возможности кастомизации среды разработки на PascalABC.NET

Идея



Текущее состояние

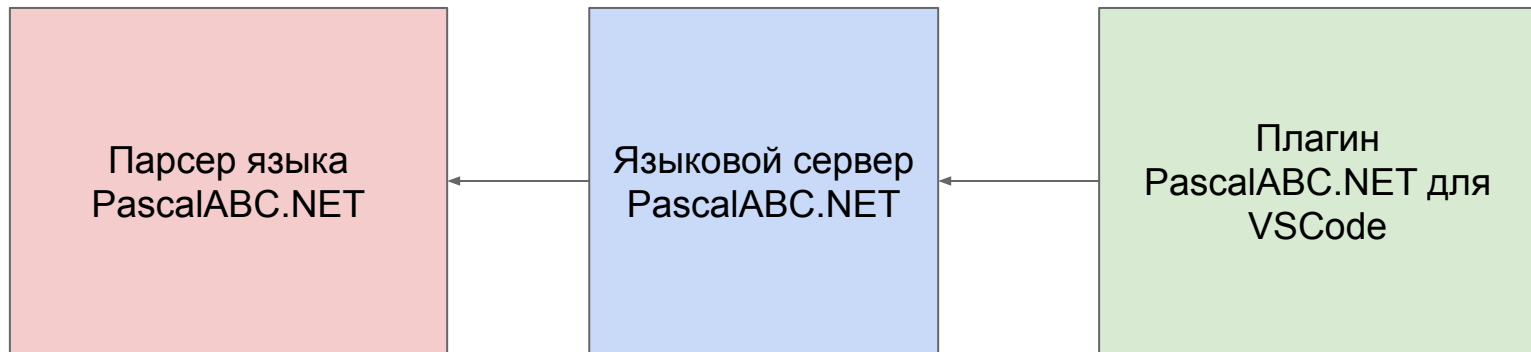
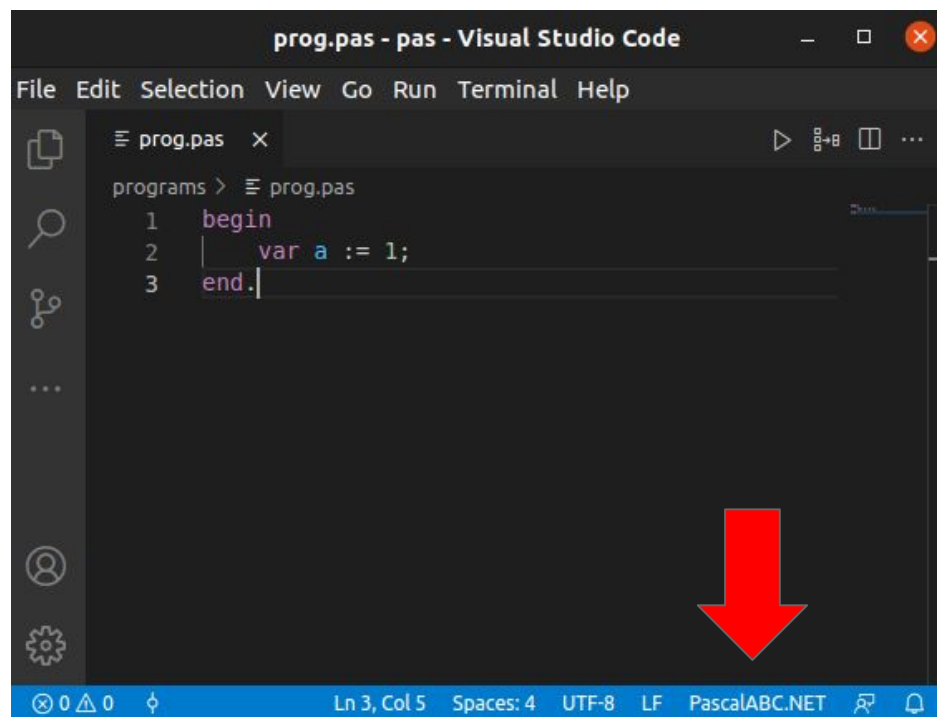
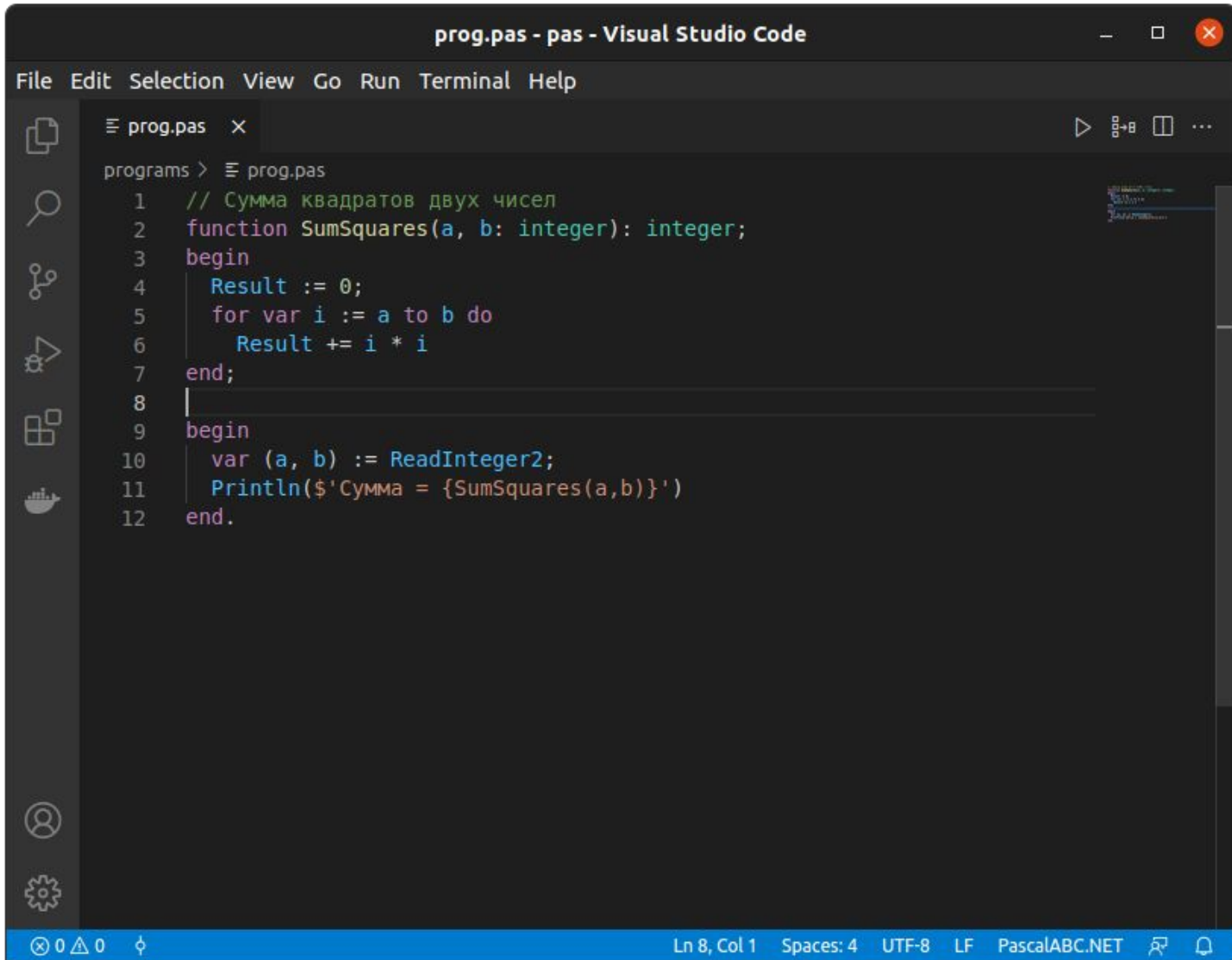


Схема работы плагина для VSCode

- Регистрация языка PascalABC.NET в редакторе при установке
- Активация расширения при открытии файла с расширением .pas
- Запуск языкового сервера и обмен информацией с ним



Подсветка синтаксиса

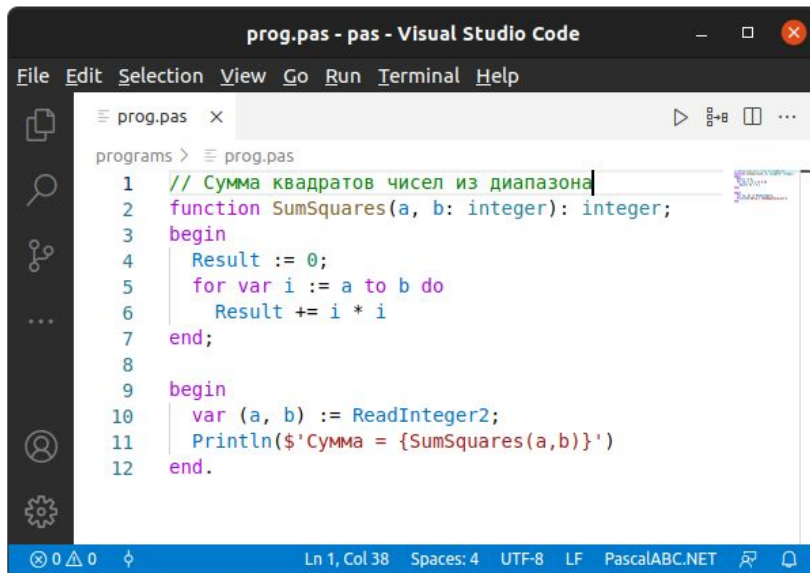


The image shows a screenshot of the Visual Studio Code editor window titled "prog.pas - pas - Visual Studio Code". The editor displays a PascalABC.NET program with syntax highlighting. The code is as follows:

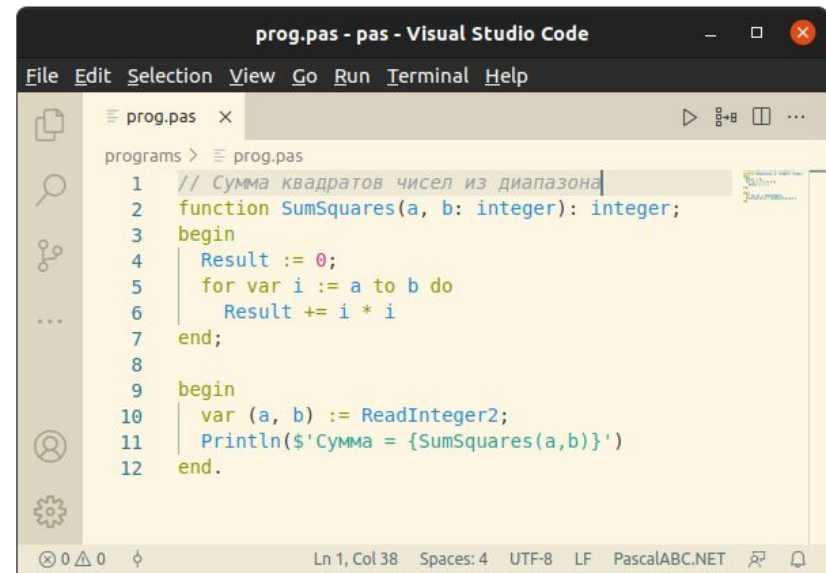
```
1  // Сумма квадратов двух чисел
2  function SumSquares(a, b: integer): integer;
3  begin
4      Result := 0;
5      for var i := a to b do
6          Result += i * i
7  end;
8
9  begin
10     var (a, b) := ReadInteger2;
11     Println('$Сумма = {SumSquares(a,b)}')
12 end.
```

The status bar at the bottom indicates the current position is "Ln 8, Col 1", with "Spaces: 4", "UTF-8", "LF", and "PascalABC.NET" encoding and line ending settings.

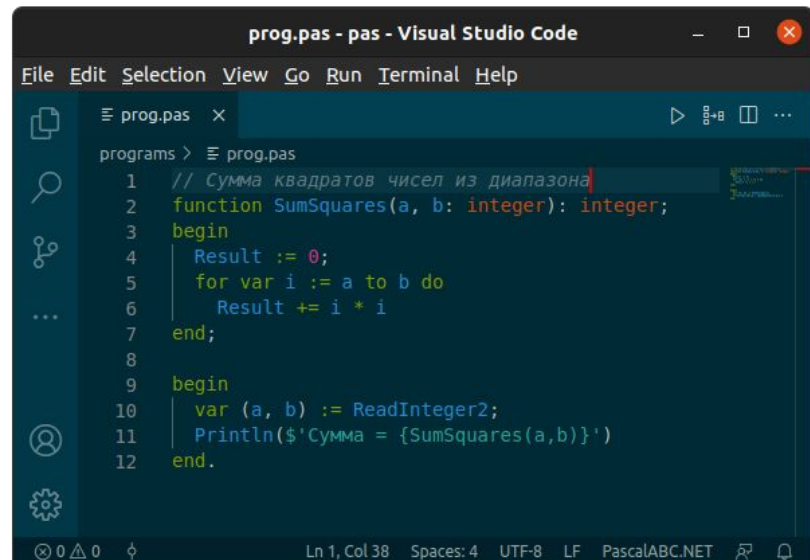
Интеграция подсветки с темами VSCode



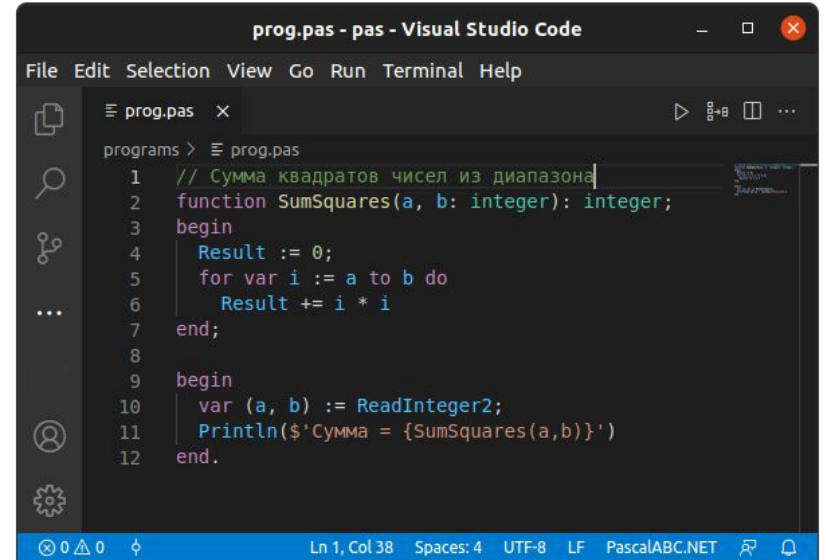
```
1 // Сумма квадратов чисел из диапазона
2 function SumSquares(a, b: integer): integer;
3 begin
4     Result := 0;
5     for var i := a to b do
6         Result += i * i
7     end;
8
9 begin
10     var (a, b) := ReadInteger2;
11     Println('$Сумма = {SumSquares(a,b)}')
12 end.
```



```
1 // Сумма квадратов чисел из диапазона
2 function SumSquares(a, b: integer): integer;
3 begin
4     Result := 0;
5     for var i := a to b do
6         Result += i * i
7     end;
8
9 begin
10     var (a, b) := ReadInteger2;
11     Println('$Сумма = {SumSquares(a,b)}')
12 end.
```



```
1 // Сумма квадратов чисел из диапазона
2 function SumSquares(a, b: integer): integer;
3 begin
4     Result := 0;
5     for var i := a to b do
6         Result += i * i
7     end;
8
9 begin
10     var (a, b) := ReadInteger2;
11     Println('$Сумма = {SumSquares(a,b)}')
12 end.
```



```
1 // Сумма квадратов чисел из диапазона
2 function SumSquares(a, b: integer): integer;
3 begin
4     Result := 0;
5     for var i := a to b do
6         Result += i * i
7     end;
8
9 begin
10     var (a, b) := ReadInteger2;
11     Println('$Сумма = {SumSquares(a,b)}')
12 end.
```

Компиляция и запуск

The screenshot displays the Visual Studio Code interface with a PascalABC.NET program being compiled and executed. The editor shows the source code for `prog.pas`, which calculates the sum of squares of integers from `a` to `b`. The terminal window shows the command to compile and run the program, followed by the output of the PascalABC.NET compiler and the execution results.

Компиляция и запуск (Compilation and Execution)

Только компиляция (Only compilation)

```
File Edit Selection View Go Run Terminal Help
prog.pas x
programs > prog.pas
1 // Сумма квадратов чисел из диапазона
2 function SumSquares(a, b: integer): integer;
3 begin
4     Result := 0;
5     for var i := a to b do
6         Result += i * i
7 end;
8
9 begin
10    var (a, b) := ReadInteger2;
11    Println($"Сумма = {SumSquares(a,b)}")
12 end.
```

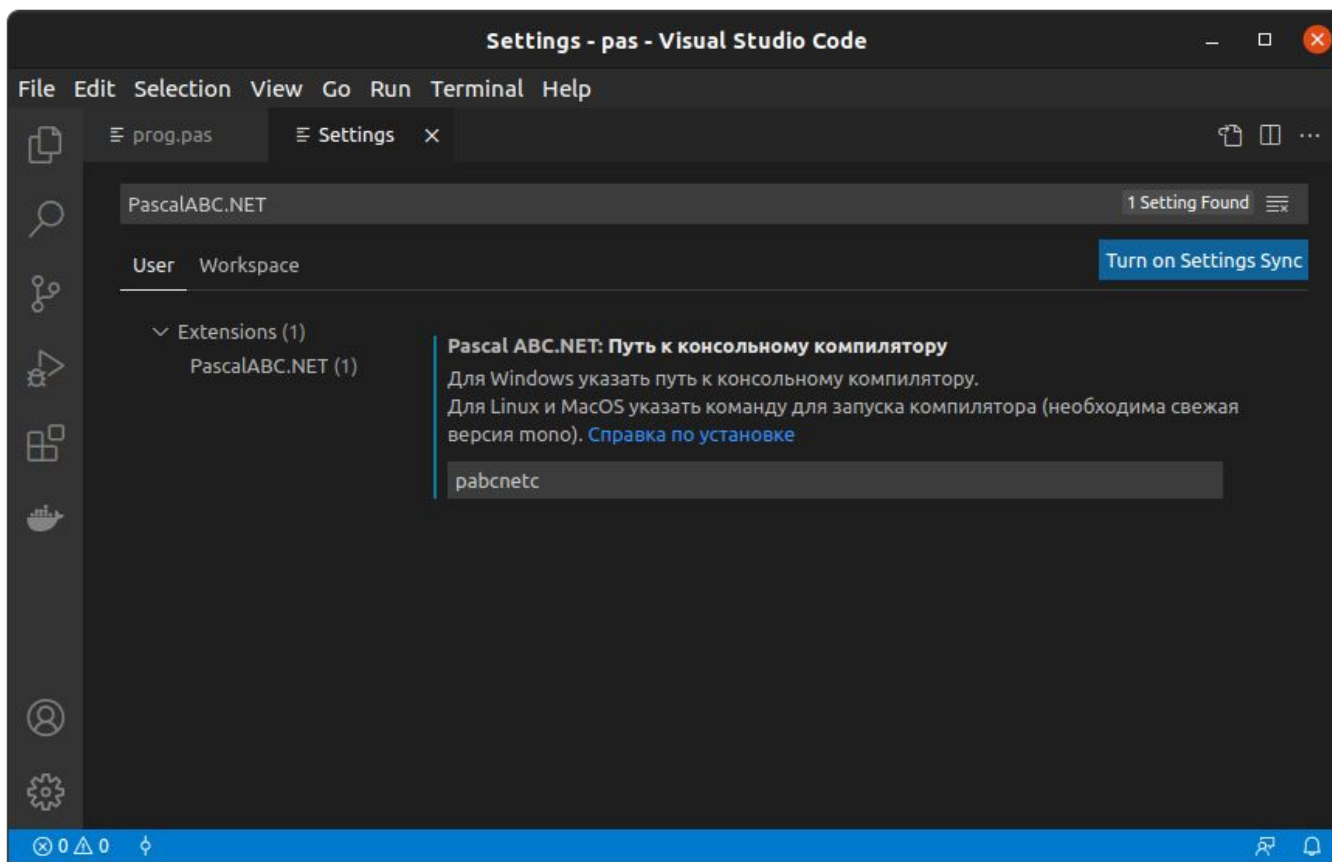
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

PascalABC.NET + - [icon] [icon] [icon]

```
lereena>pabcnetc "/home/kostrikovan/pepe/pas/programs/prog.pas"; mono "/home/kostrikovan/pepe/p
as/programs/prog.exe"
Loading core...
Подключен парсер PascalABC.NET Language Parser v1.2
Подключен парсер Documentation Comments Tag Parser v0.9
PascalABCCompiler.Core v3.8.0.2932
Copyright (c) 2005-2021 by Ivan Bondarev, Stanislav Mikhalkovich
OK 90.416ms
Connected parsers: PascalABC.NET (*.pas);
Connected conversions: Optimizer;
Compiling assembly prog.pas...
OK 1027.68ms (11lines)
10 15
Сумма = 955
lereena>
```

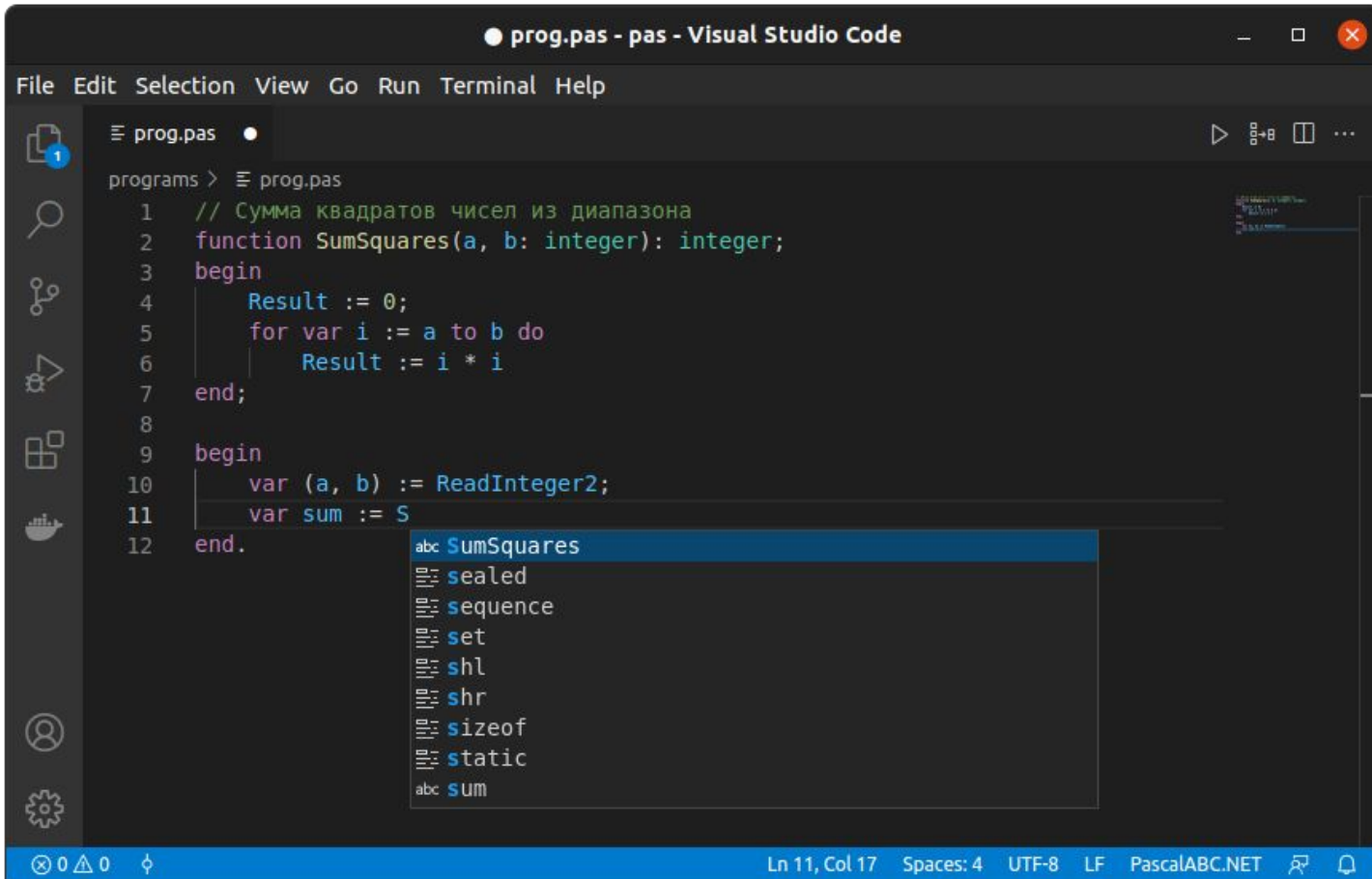
Ln 12, Col 5 Spaces: 4 UTF-8 LF PascalABC.NET

Настройка компилятора



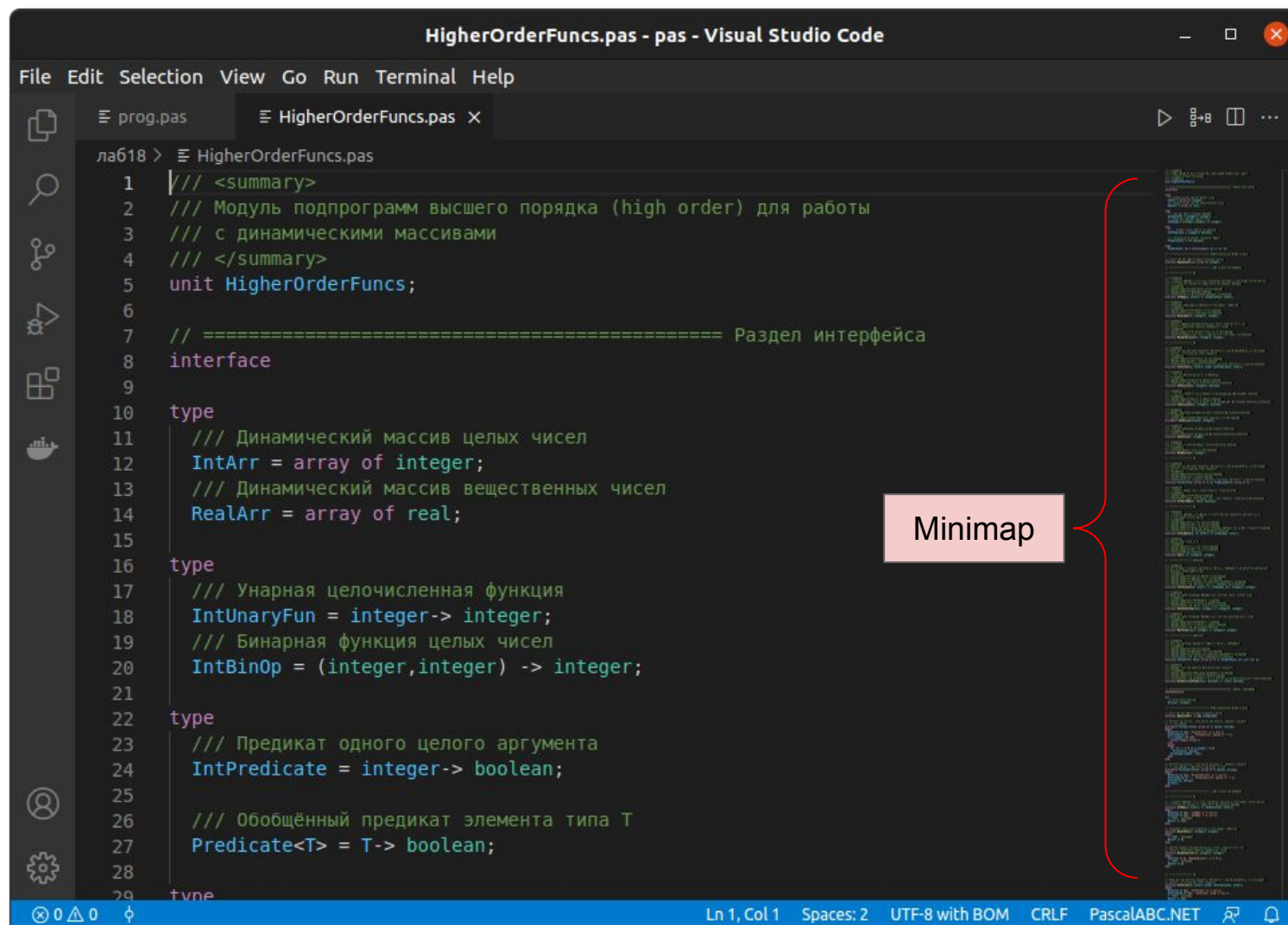
Пакет автоматической установки компилятора для некоторых дистрибутивов Linux: <https://github.com/COOLIRON2311/pabcnetdeb> (Автор: Иван Игнатенко)

Автодополнение



Несколько полезных возможностей VSCode

Упрощённая навигация по коду



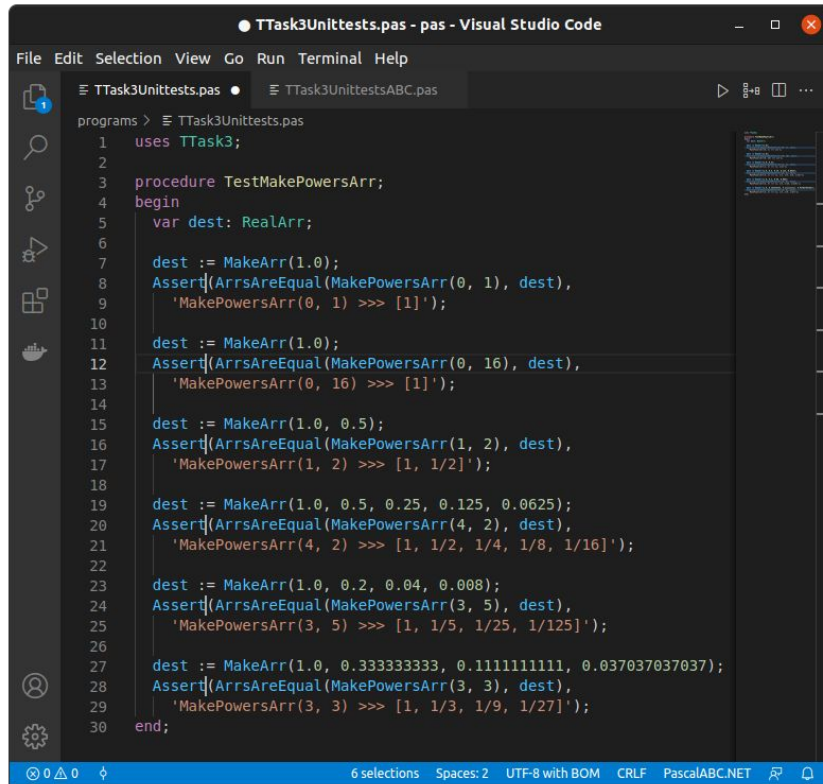
Задача: перевести существующие тесты на NUnitABC

```
TTask3UnitTests.pas - pas - Visual Studio Code
File Edit Selection View Go Run Terminal Help
TTask3UnitTests.pas x TTask3UnitTestsABC.pas
programs > TTask3UnitTests.pas
1 uses TTask3;
2
3 procedure TestMakePowersArr;
4 begin
5   var dest: RealArr;
6
7   dest := MakeArr(1.0);
8   Assert(ArrsAreEqual(MakePowersArr(0, 1), dest),
9     'MakePowersArr(0, 1) >>> [1]');
10
11   dest := MakeArr(1.0);
12   Assert(ArrsAreEqual(MakePowersArr(0, 16), dest),
13     'MakePowersArr(0, 16) >>> [1]');
14
15   dest := MakeArr(1.0, 0.5);
16   Assert(ArrsAreEqual(MakePowersArr(1, 2), dest),
17     'MakePowersArr(1, 2) >>> [1, 1/2]');
18
19   dest := MakeArr(1.0, 0.5, 0.25, 0.125, 0.0625);
20   Assert(ArrsAreEqual(MakePowersArr(4, 2), dest),
21     'MakePowersArr(4, 2) >>> [1, 1/2, 1/4, 1/8, 1/16]');
22
23   dest := MakeArr(1.0, 0.2, 0.04, 0.008);
24   Assert(ArrsAreEqual(MakePowersArr(3, 5), dest),
25     'MakePowersArr(3, 5) >>> [1, 1/5, 1/25, 1/125]');
26
27   dest := MakeArr(1.0, 0.333333333, 0.111111111, 0.037037037);
28   Assert(ArrsAreEqual(MakePowersArr(3, 3), dest),
29     'MakePowersArr(3, 3) >>> [1, 1/3, 1/9, 1/27]');
30 end;
```

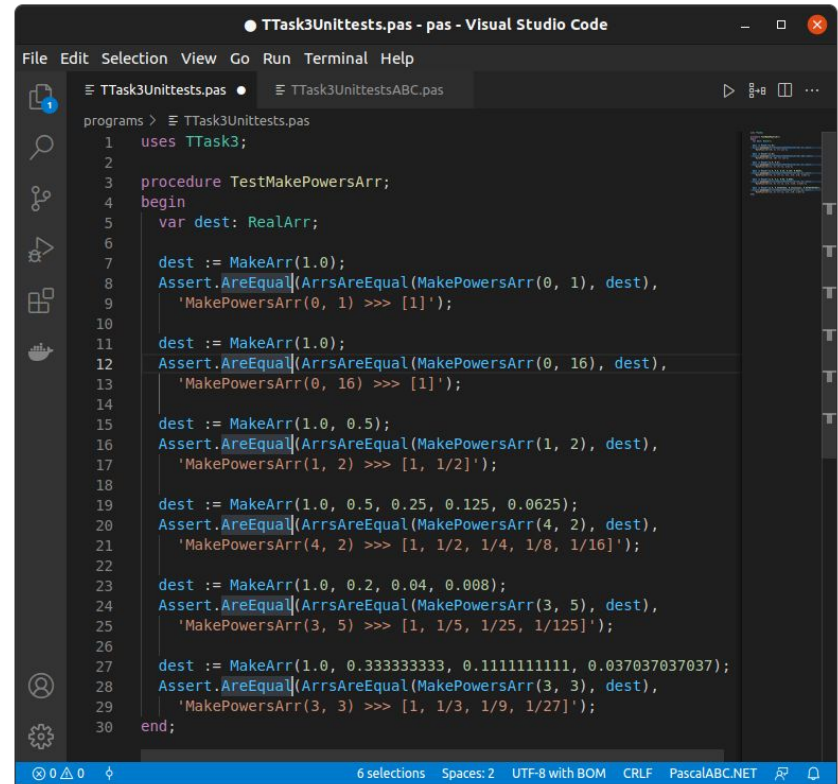


```
TTask3UnitTests.pas - pas - Visual Studio Code
File Edit Selection View Go Run Terminal Help
TTask3UnitTests.pas x TTask3UnitTestsABC.pas
programs > TTask3UnitTests.pas
1 uses NUnitABC;
2 uses TTask3;
3
4 [Test]
5 procedure TestMakePowersArr;
6 begin
7   var dest: RealArr;
8
9   dest := MakeArr(1.0);
10  Assert.AreEqual(MakePowersArr(0, 1), dest),
11    'MakePowersArr(0, 1) >>> [1]');
12
13  dest := MakeArr(1.0);
14  Assert.AreEqual(MakePowersArr(0, 16), dest),
15    'MakePowersArr(0, 16) >>> [1]');
16
17  dest := MakeArr(1.0, 0.5);
18  Assert.AreEqual(MakePowersArr(1, 2), dest),
19    'MakePowersArr(1, 2) >>> [1, 1/2]');
20
21  dest := MakeArr(1.0, 0.5, 0.25, 0.125, 0.0625);
22  Assert.AreEqual(MakePowersArr(4, 2), dest),
23    'MakePowersArr(4, 2) >>> [1, 1/2, 1/4, 1/8, 1/16]');
24
25  dest := MakeArr(1.0, 0.2, 0.04, 0.008);
26  Assert.AreEqual(MakePowersArr(3, 5), dest),
27    'MakePowersArr(3, 5) >>> [1, 1/5, 1/25, 1/125]');
28
29  dest := MakeArr(1.0, 0.333333333, 0.111111111, 0.037037037);
30  Assert.AreEqual(MakePowersArr(3, 3), dest),
31    'MakePowersArr(3, 3) >>> [1, 1/3, 1/9, 1/27]');
32 end;
```

Мультикурсор (Alt + клик)



```
1 uses TTask3;  
2  
3 procedure TestMakePowersArr;  
4 begin  
5   var dest: RealArr;  
6  
7   dest := MakeArr(1.0);  
8   Assert(ArrsAreEqual(MakePowersArr(0, 1), dest),  
9     'MakePowersArr(0, 1) >>> [1]');  
10  
11   dest := MakeArr(1.0);  
12   Assert(ArrsAreEqual(MakePowersArr(0, 16), dest),  
13     'MakePowersArr(0, 16) >>> [1]');  
14  
15   dest := MakeArr(1.0, 0.5);  
16   Assert(ArrsAreEqual(MakePowersArr(1, 2), dest),  
17     'MakePowersArr(1, 2) >>> [1, 1/2]');  
18  
19   dest := MakeArr(1.0, 0.5, 0.25, 0.125, 0.0625);  
20   Assert(ArrsAreEqual(MakePowersArr(4, 2), dest),  
21     'MakePowersArr(4, 2) >>> [1, 1/2, 1/4, 1/8, 1/16]');  
22  
23   dest := MakeArr(1.0, 0.2, 0.04, 0.008);  
24   Assert(ArrsAreEqual(MakePowersArr(3, 5), dest),  
25     'MakePowersArr(3, 5) >>> [1, 1/5, 1/25, 1/125]');  
26  
27   dest := MakeArr(1.0, 0.333333333, 0.111111111, 0.037037037);  
28   Assert(ArrsAreEqual(MakePowersArr(3, 3), dest),  
29     'MakePowersArr(3, 3) >>> [1, 1/3, 1/9, 1/27]');  
30 end;
```



```
1 uses TTask3;  
2  
3 procedure TestMakePowersArr;  
4 begin  
5   var dest: RealArr;  
6  
7   dest := MakeArr(1.0);  
8   Assert(ArrsAreEqual(MakePowersArr(0, 1), dest),  
9     'MakePowersArr(0, 1) >>> [1]');  
10  
11   dest := MakeArr(1.0);  
12   Assert(ArrsAreEqual(MakePowersArr(0, 16), dest),  
13     'MakePowersArr(0, 16) >>> [1]');  
14  
15   dest := MakeArr(1.0, 0.5);  
16   Assert(ArrsAreEqual(MakePowersArr(1, 2), dest),  
17     'MakePowersArr(1, 2) >>> [1, 1/2]');  
18  
19   dest := MakeArr(1.0, 0.5, 0.25, 0.125, 0.0625);  
20   Assert(ArrsAreEqual(MakePowersArr(4, 2), dest),  
21     'MakePowersArr(4, 2) >>> [1, 1/2, 1/4, 1/8, 1/16]');  
22  
23   dest := MakeArr(1.0, 0.2, 0.04, 0.008);  
24   Assert(ArrsAreEqual(MakePowersArr(3, 5), dest),  
25     'MakePowersArr(3, 5) >>> [1, 1/5, 1/25, 1/125]');  
26  
27   dest := MakeArr(1.0, 0.333333333, 0.111111111, 0.037037037);  
28   Assert(ArrsAreEqual(MakePowersArr(3, 3), dest),  
29     'MakePowersArr(3, 3) >>> [1, 1/3, 1/9, 1/27]');  
30 end;
```

Публикация плагина

 Open VSX Registry

Documentation FAQ Community About 

**pascalabcnet**
✓ lereenadem.pascalabcnet | Published by Lereena  | MIT
PascalABC.NET language extension
1 download | ☆☆☆☆☆(0)

OVERVIEW CHANGES RA

PascalABC.NET VSCode extension

Install 

The extension is published in Visual Studio Marketplace

program.pas - pas - Visual Studio Code
File Edit Selection View Go Run Terminal Help
F program.pas X
1 function SumSquares(a, b: integer): integer;
2 begin
3 Result := 0;

VisualStudio | Marketplace Sign out 

Visual Studio Code > Other > pascalabcnet New to Visual Studio Code? [Get it now.](#)

**pascalabcnet**
lereenadem | 643 installs | ☆☆☆☆☆(0) | Free
PascalABC.NET language extension
Installation
Launch VS Code Quick Open (Ctrl+P), paste the following command, and press enter.
 [More Info](#)

Overview Version History Q & A Rating & Review

PascalABC.NET VSCode extension

Install

The extension is published in Visual Studio Marketplace and Eclipse Open VSX and can be installed from these registries.

program.pas - pas - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXTENSIONS MARKETPLACE
PascalABC.NET
F program.pas X
F program.pas

Categories
Other

Tags
PascalABC.NET pascalabcnet

Works with
Universal

Resources

Технические моменты

- Актуальная версия: **1.1.3**
- Требуется версия VSCode не ниже **1.62.0**
- Протестирован под Linux Ubuntu 21.04, Manjaro Linux и Windows 10

Где скачать?

- В меню *Extensions (Расширения)* VSCode
- Visual Studio Marketplace:
<https://marketplace.visualstudio.com/items?itemName=lereenadem.pascalabcnet>
- Open VSX Registry:
<https://open-vsx.org/extension/lereenadem/pascalabcnet>
- Исходный код на GitHub:
<https://github.com/Lereena/pascalabcnet-vs>