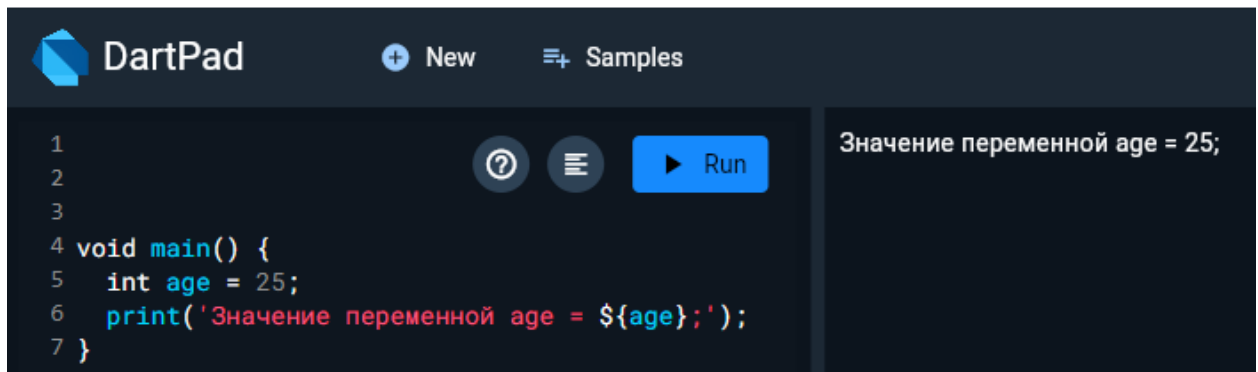


Задание № 1: Объявление и инициализация переменной типа `int`.

Объявить переменную `age` типа `int` и присвоить ей значение 25.

Описание алгоритма решения:

1. Объявляем переменную `age` типа `int`.
2. Присваиваем ей значение `25`.
3. Выводим значение переменной `age` на экран с помощью функции `print()`.



```
1
2
3
4 void main() {
5   int age = 25;
6   print('Значение переменной age = ${age}');
7 }
```

Значение переменной age = 25;

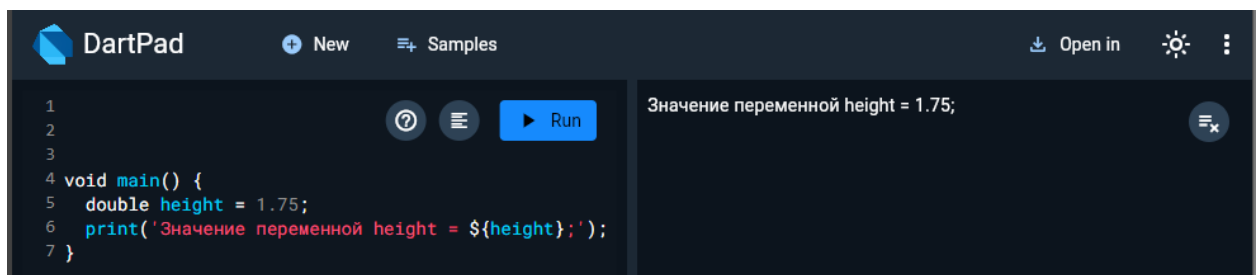
Рисунок №1 – Результат решения задания №1.

Задание № 2: Объявление и инициализация переменной типа `double`

Объявить переменную `height` типа `double` и присвоить ей значение `1.75`.

Описание алгоритма решения:

1. Объявляем переменную `height` типа `double`.
2. Присваиваем ей значение `1.75`.
3. Выводим значение переменной `height` на экран с помощью функции `print()`.



```
1
2
3
4 void main() {
5   double height = 1.75;
6   print('Значение переменной height = ${height}');
7 }
```

Значение переменной height = 1.75;

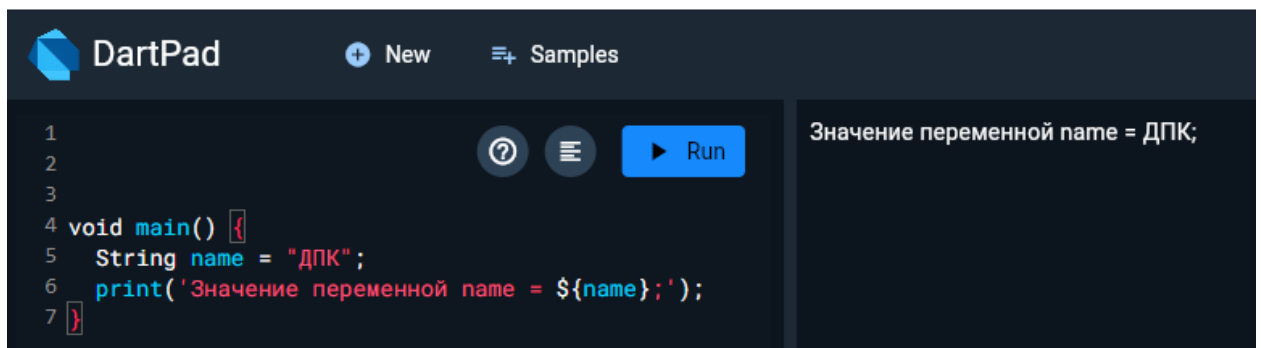
Рисунок №2 – Результат решения задания №2.

Задание № 3: Объявление и инициализация переменной типа `String`

Объявить переменную *name* типа *String* и присвоить ей значение «ДПК».

Описание алгоритма решения:

1. Объявляем переменную *name* типа *String*.
2. Присваиваем ей значение «ДПК».
3. Выводим значение переменной *name* на экран с помощью функции *print()*.



```
1
2
3
4 void main() {
5   String name = "ДПК";
6   print('Значение переменной name = ${name}');
7 }
```

Значение переменной name = ДПК;

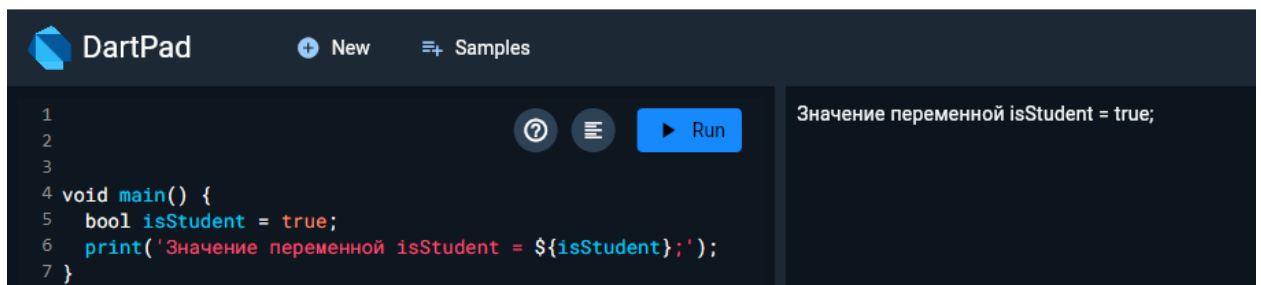
Рисунок №3 – Результат решения задания №3.

Задание № 4: Объявление и инициализация переменной типа `bool`

Объявить переменную *isStudent* типа *bool* и присвоить ей значение *true*.

Описание алгоритма решения:

1. Объявляем переменную *isStudent* типа *bool*.
2. Присваиваем ей значение *true*.
3. Выводим значение переменной *isStudent* на экран с помощью функции *print()*.



```
1
2
3
4 void main() {
5   bool isStudent = true;
6   print('Значение переменной isStudent = ${isStudent}');
7 }
```

Значение переменной isStudent = true;

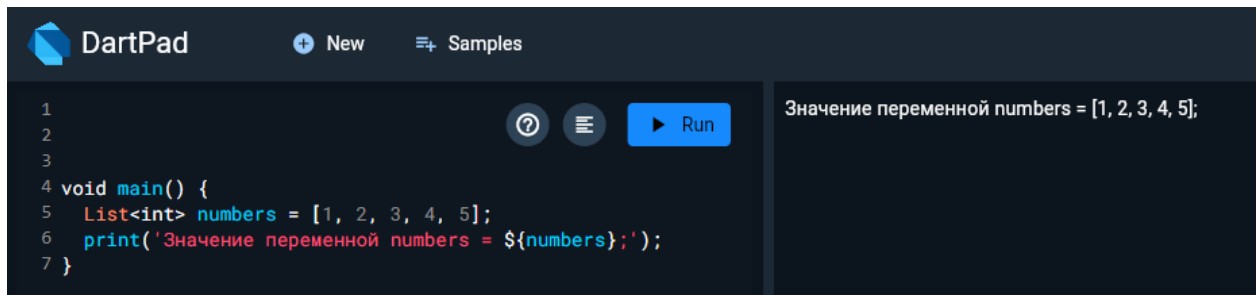
Рисунок №4 – Результат решения задания №4.

Задание № 5: Объявление и инициализация переменной типа List

Объявить переменную `numbers` типа `List<int>` и присвоить ей список чисел `[1, 2, 3, 4, 5]`.

Описание алгоритма решения:

1. Объявляем переменную `numbers` типа `List<int>`.
2. Присваиваем ей список чисел `[1, 2, 3, 4, 5]`.
3. Выводим значение переменной `numbers` на экран с помощью функции `print()`.



```
1
2
3
4 void main() {
5   List<int> numbers = [1, 2, 3, 4, 5];
6   print('Значение переменной numbers = ${numbers}');
7 }
```

Значение переменной numbers = [1, 2, 3, 4, 5];

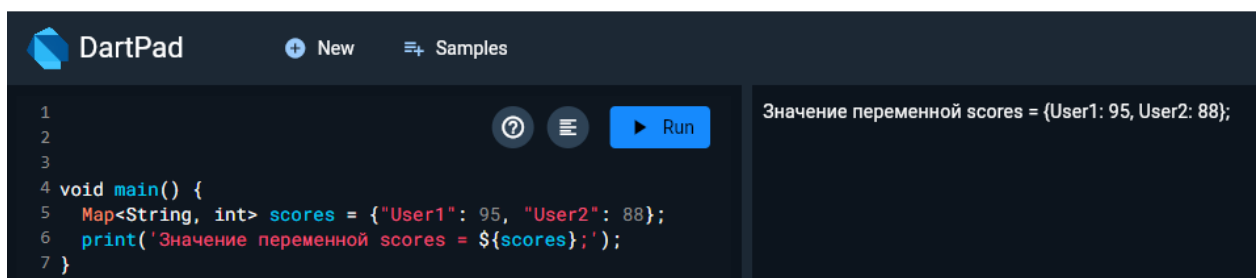
Рисунок №5 – Результат решения задания №5.

Задание № 6: Объявление и инициализация переменной типа Map

Объявить переменную `scores` типа `Map<String, int>` и присвоить ей словарь `{"User1": 95, "User2": 88}`.

Описание алгоритма решения:

1. Объявляем переменную `scores` типа `Map<String, int>`.
2. Присваиваем ей словарь `{"User1": 95, "User2": 88}`.
3. Выводим значение переменной `scores` на экран с помощью функции `print()`.



```
1
2
3
4 void main() {
5   Map<String, int> scores = {"User1": 95, "User2": 88};
6   print('Значение переменной scores = ${scores}');
7 }
```

Значение переменной scores = {User1: 95, User2: 88};

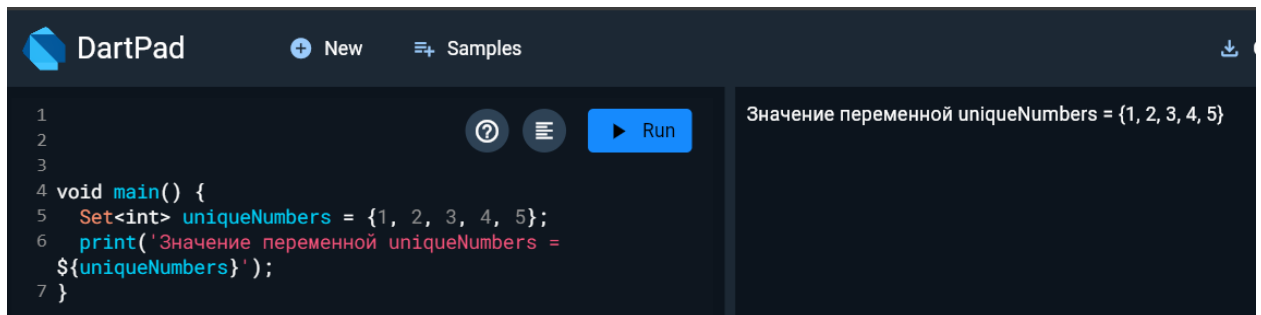
Рисунок №6 – Результат решения задания №6.

Задание № 7: Объявление и инициализация переменной типа Set

Объявить переменную uniqueNumbers типа Set<int> и присвоить ей множество {1, 2, 3, 4, 5}.

Описание алгоритма решения:

1. Объявляем переменную uniqueNumbers типа Set<int>.
2. Присваиваем ей множество {1, 2, 3, 4, 5}.
3. Выводим значение переменной uniqueNumbers на экран с помощью функции print().



```
1
2
3
4 void main() {
5   Set<int> uniqueNumbers = {1, 2, 3, 4, 5};
6   print('Значение переменной uniqueNumbers =
7   ${uniqueNumbers}');
8 }
```

Значение переменной uniqueNumbers = {1, 2, 3, 4, 5}

Рисунок №7 – Результат решения задания №7.

Задание № 8: Объявление и инициализация переменной типа Runes

Объявить переменную emoji типа Runes и присвоить ей символ с кодом 0x1F600 (улыбающийся смайлик).

Описание алгоритма решения:

1. Объявляем переменную emoji типа Runes.
2. Присваиваем ей символ с кодом 0x1F600.
3. Преобразуем Runes в строку с помощью String.fromCharCode и выводим на экран с помощью функции print().



```
1
2
3
4 void main() {
5   Runes emoji = Runes('\u{1F600}');
6   print('Значение переменной emoji =
7   ${String.fromCharCode(emoji)}');
8 }
```

Значение переменной emoji = 😊

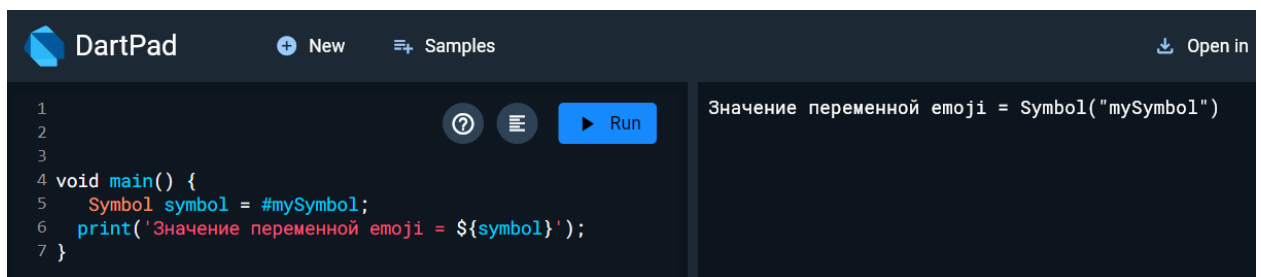
Рисунок №8 – Результат решения задания №8.

Задание № 9: Объявление и инициализация переменной типа `Symbol`

Объявить переменную `symbol` типа `Symbol` и присвоить ей символ `#mySymbol`.

Описание алгоритма решения:

1. Объявляем переменную `symbol` типа `Symbol`.
2. Присваиваем ей символ `#mySymbol`.
3. Выводим значение переменной `symbol` на экран с помощью функции `print()`.



```
1
2
3
4 void main() {
5   Symbol symbol = #mySymbol;
6   print('Значение переменной emoji = ${symbol}');
7 }
```

Значение переменной emoji = Symbol("mySymbol")

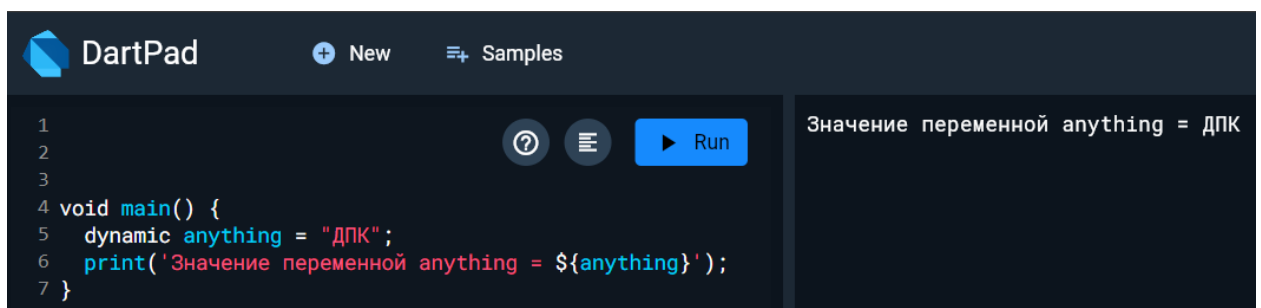
Рисунок №9 – Результат решения задания №9.

Задание № 10: Объявление и инициализация переменной типа `dynamic`

Объявить переменную `anything` типа `dynamic` и присвоить ей значение «ДПК».

Описание алгоритма решения:

1. Объявляем переменную `anything` типа `dynamic`.
2. Присваиваем ей значение «ДПК».
3. Выводим значение переменной `anything` на экран с помощью функции `print()`.



```
1
2
3
4 void main() {
5   dynamic anything = "ДПК";
6   print('Значение переменной anything = ${anything}');
7 }
```

Значение переменной anything = ДПК

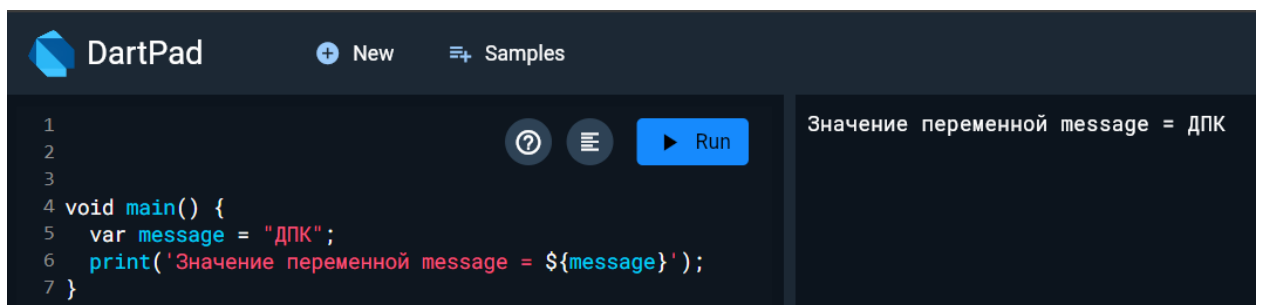
Рисунок №10 – Результат решения задания №10.

Задание № 11: Объявление и инициализация переменной типа `var`

Объявить переменную `message` с помощью `var` и присвоить ей значение «ДПК».

Описание алгоритма решения:

1. Объявляем переменную `message` с помощью `var`.
2. Присваиваем ей значение «ДПК».
3. Выводим значение переменной `message` на экран с помощью функции `print()`.



```
1
2
3
4 void main() {
5   var message = "ДПК";
6   print('Значение переменной message = ${message}');
7 }
```

Значение переменной message = ДПК

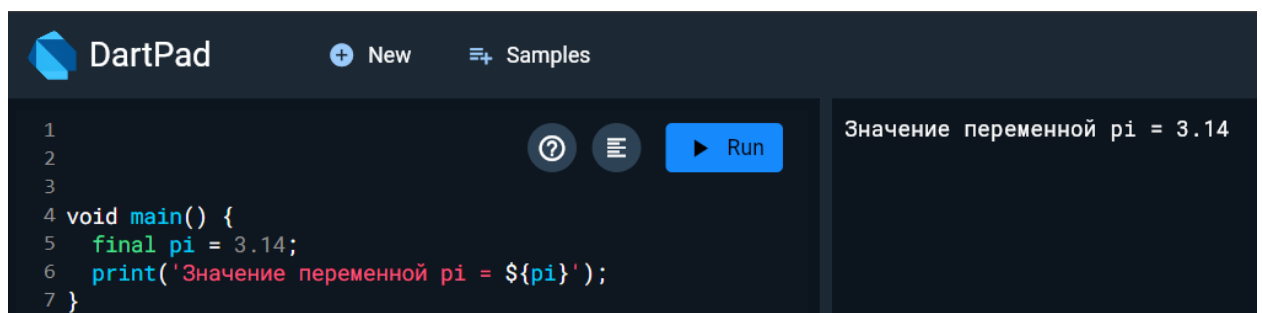
Рисунок №11 – Результат решения задания №11.

Задание № 12: Объявление и инициализация переменной типа `final`

Объявить переменную `pi` типа `final` и присвоить ей значение `3.14`.

Описание алгоритма решения:

1. Объявляем переменную `pi` типа `final`.
2. Присваиваем ей значение `3.14`.
3. Выводим значение переменной `pi` на экран с помощью функции `print()`.



```
1
2
3
4 void main() {
5   final pi = 3.14;
6   print('Значение переменной pi = ${pi}');
7 }
```

Значение переменной pi = 3.14

Рисунок №12 – Результат решения задания №12.

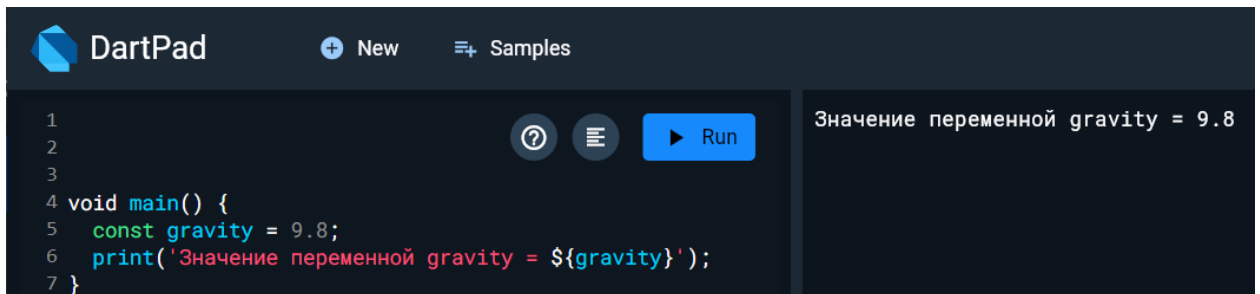
Задание № 13: Объявление и инициализация переменной

типа `const`

Объявить переменную *gravity* типа *const* и присвоить ей значение 9.8.

Описание алгоритма решения:

1. Объявляем переменную *gravity* типа *const*.
2. Присваиваем ей значение 9.8.
3. Выводим значение переменной *gravity* на экран с помощью функции *print()*.



```
1
2
3
4 void main() {
5   const gravity = 9.8;
6   print('Значение переменной gravity = ${gravity}');
7 }
```

Значение переменной gravity = 9.8

Рисунок №13 – Результат решения задания №13.

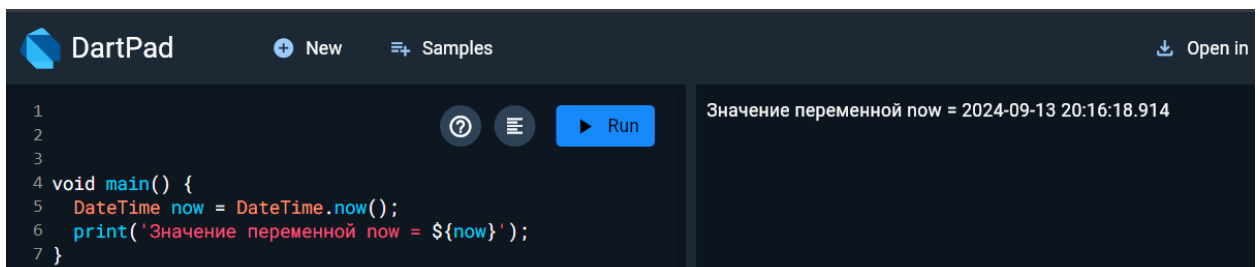
Задание № 14: Объявление и инициализация переменной

типа `DateTime`

Объявить переменную *now* типа *DateTime* и присвоить ей *текущую дату и время*.

Описание алгоритма решения:

1. Объявляем переменную *now* типа *DateTime*.
2. Присваиваем ей *текущую дату и время* с помощью *DateTime.now()*.
3. Выводим значение переменной *now* на экран с помощью функции *print()*.



```
1
2
3
4 void main() {
5   DateTime now = DateTime.now();
6   print('Значение переменной now = ${now}');
7 }
```

Значение переменной now = 2024-09-13 20:16:18.914

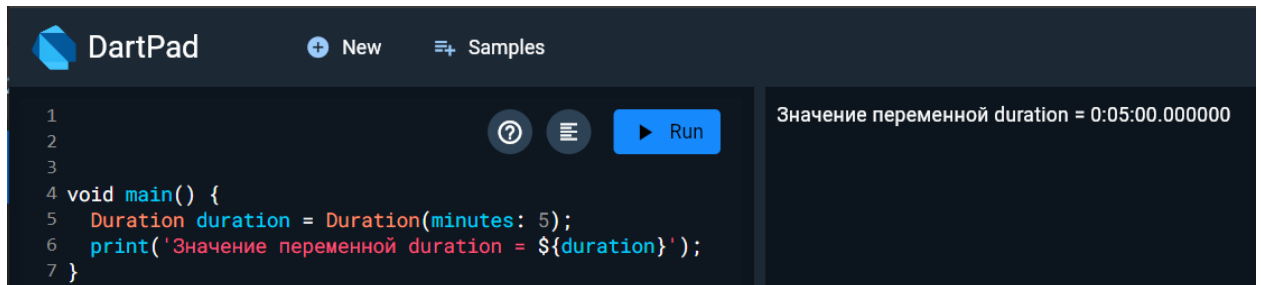
Рисунок №14 – Результат решения задания №14.

Задание № 15: Объявление и инициализация переменной типа `Duration`

Объявить переменную `duration` типа `Duration` и присвоить ей продолжительность 5 минут.

Описание алгоритма решения:

1. Объявляем переменную `duration` типа `Duration`.
2. Присваиваем ей продолжительность 5 минут с помощью `Duration(minutes: 5)`.
3. Выводим значение переменной `duration` на экран с помощью функции `print()`.



```
1
2
3
4 void main() {
5   Duration duration = Duration(minutes: 5);
6   print('Значение переменной duration = ${duration}');
7 }
```

Значение переменной duration = 0:05:00.000000

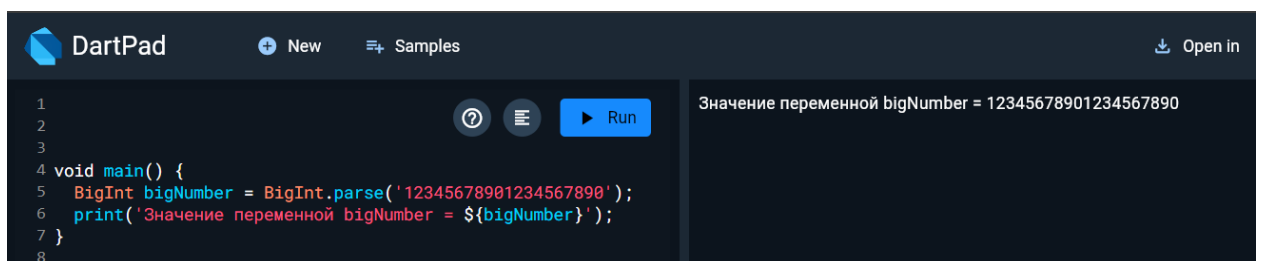
Рисунок №15 – Результат решения задания №15.

Задание № 16: Объявление и инициализация переменной типа `BigInt`

Объявить переменную `bigNumber` типа `BigInt` и присвоить ей значение 12345678901234567890.

Описание алгоритма решения:

1. Объявляем переменную `bigNumber` типа `BigInt`.
2. Присваиваем ей значение 12345678901234567890 с помощью `BigInt.parse()`.
3. Выводим значение переменной `bigNumber` на экран с помощью функции `print()`.



```
1
2
3
4 void main() {
5   BigInt bigNumber = BigInt.parse('12345678901234567890');
6   print('Значение переменной bigNumber = ${bigNumber}');
7 }
8
```

Значение переменной bigNumber = 12345678901234567890

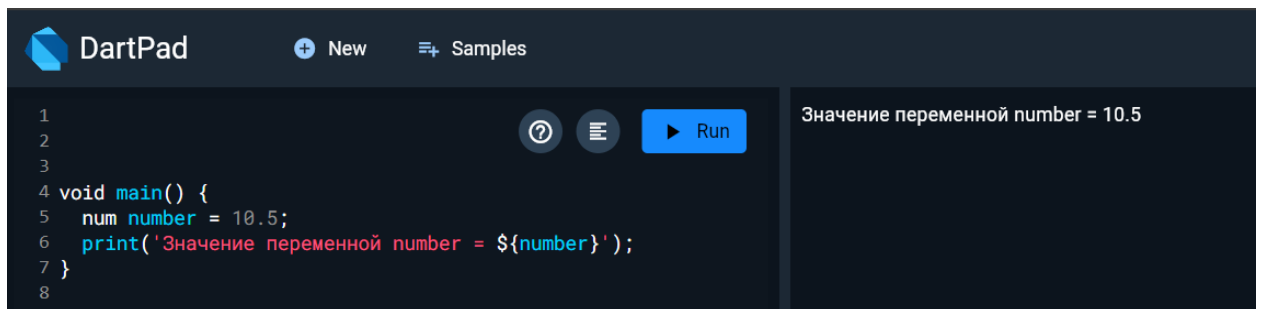
Рисунок №16 – Результат решения задания №16.

Задание № 17: Объявление и инициализация переменной типа `num`

Объявить переменную `number` типа `num` и присвоить ей значение `10.5`.

Описание алгоритма решения:

1. Объявляем переменную `number` типа `num`.
2. Присваиваем ей значение `10.5`.
3. Выводим значение переменной `number` на экран с помощью функции `print()`.



```
1
2
3
4 void main() {
5   num number = 10.5;
6   print('Значение переменной number = ${number}');
7 }
8
```

Значение переменной number = 10.5

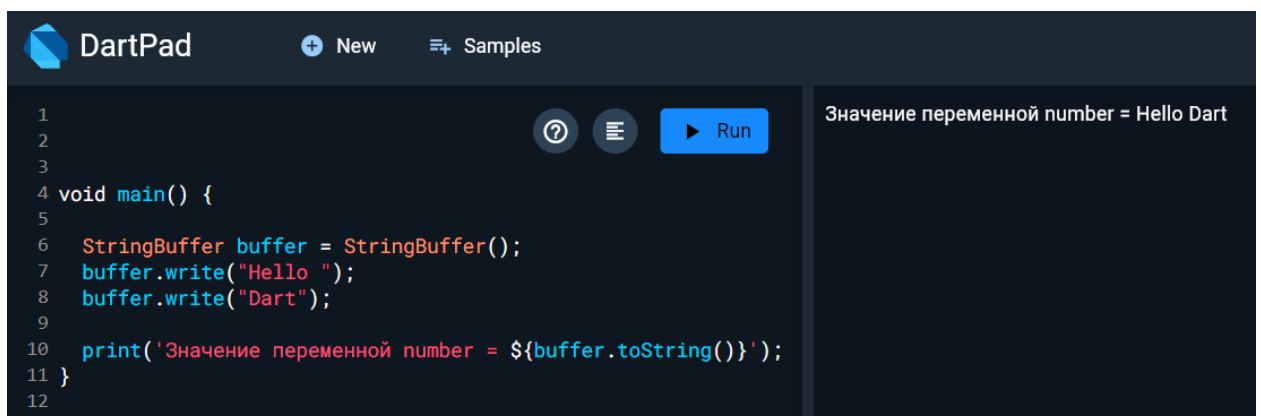
Рисунок №17 – Результат решения задания №17.

Задание № 18: Объявление и инициализация переменной типа `StringBuffer`

Объявить переменную `buffer` типа `StringBuffer` и добавить в нее строки «`Hello`» и «`Dart`».

Описание алгоритма решения:

1. Объявляем переменную `buffer` типа `StringBuffer`.
2. Добавляем строку «`Hello`» с помощью `buffer.write`.
3. Добавляем строку «`Dart`» с помощью `buffer.write`.
4. Преобразуем `StringBuffer` в строку с помощью `buffer.toString()` и выводим на экран функции `print()`.



```
1
2
3
4 void main() {
5
6   StringBuffer buffer = StringBuffer();
7   buffer.write("Hello ");
8   buffer.write("Dart");
9
10  print('Значение переменной number = ${buffer.toString()}');
11 }
12
```

Значение переменной number = Hello Dart

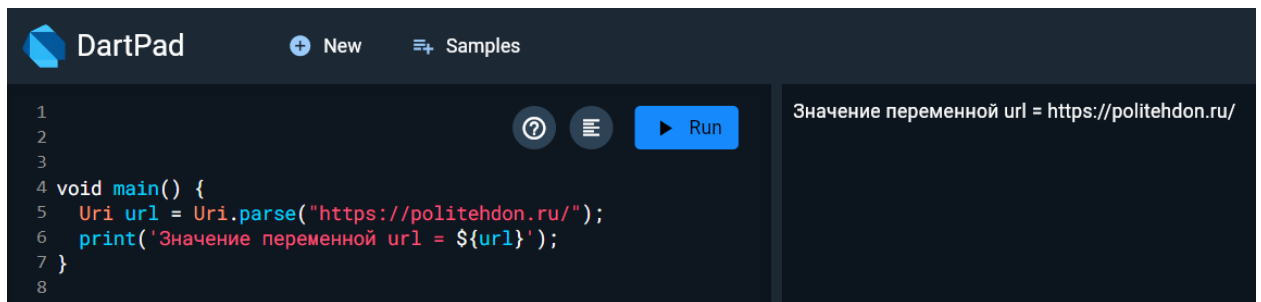
Рисунок №18 – Результат решения задания №18.

Задание № 19: Объявление и инициализация переменной типа Uri

Объявить переменную url типа Uri и присвоить ей URL «<https://politehdon.ru/>».

Описание алгоритма решения:

1. Объявляем переменную url типа Uri.
2. Присваиваем ей URL «<https://politehdon.ru/>» с помощью Uri.parse.
3. Выводим значение переменной url на экран с помощью функции print().



```
1
2
3
4 void main() {
5   Uri url = Uri.parse("https://politehdon.ru/");
6   print('Значение переменной url = ${url}');
7 }
8
```

Значение переменной url = https://politehdon.ru/

Рисунок №19 – Результат решения задания №19.

Индивидуальные задания для закрепления материала

Простые операции:

1. **Калькулятор:** Создайте консольное приложение-калькулятор, которое может выполнять основные арифметические операции (сложение, вычитание, умножение, деление) с двумя числами. Используйте типы данных int и double.
2. **Конвертер температуры:** Напишите программу, которая конвертирует температуру из градусов Цельсия в градусы Фаренгейта и наоборот. Используйте тип данных double.
3. **Генератор случайных чисел:** Создайте программу, которая генерирует случайное число в заданном диапазоне. Используйте тип данных int и класс Random.
4. **Проверка на четность:** Напишите функцию, которая принимает целое число и возвращает true, если число четное, и false, если нечетное. Используйте тип данных bool.
5. **Сумма цифр числа:** Напишите программу, которая вычисляет сумму цифр заданного целого числа. Используйте тип данных int.

Работа со строками:

1. **Подсчет символов:** Напишите программу, которая подсчитывает количество символов в строке. Используйте тип данных String.
2. **Поиск подстроки:** Напишите программу, которая проверяет, содержится ли подстрока в заданной строке. Используйте метод contains().
3. **Замена подстроки:** Напишите программу, которая заменяет все вхождения подстроки в строке на другую строку. Используйте метод replaceAll().
4. **Обратная строка:** Напишите программу, которая переворачивает заданную строку. Используйте метод split(), reversed и join().
5. **Проверка на палиндром:** Напишите программу, которая проверяет, является ли заданная строка палиндромом (читается одинаково слева направо и справа налево). Используйте метод toLowerCase() и split(), reversed, join().

Работа с коллекциями:

1. **Список чисел:** Создайте список целых чисел и выполните над ним следующие операции: добавьте элемент, удалите элемент, найдите максимальное и минимальное значение. Используйте тип данных *List<int>*.
2. **Уникальные элементы:** Напишите программу, которая удаляет все дубликаты из списка. Используйте тип данных *Set*.
3. **Сортировка списка:** Напишите программу, которая сортирует список строк по алфавиту. Используйте метод *sort()*.
4. **Словарь:** Создайте словарь, который хранит пары «ключ-значение» (например, «имя» - «возраст»). Добавьте, удалите и найдите элементы в словаре. Используйте тип данных *Map<String, dynamic>*.
5. **Сумма элементов списка:** Напишите программу, которая вычисляет сумму всех элементов в списке чисел. Используйте метод *reduce()*.

Продвинутые задания:

1. **Игра «Угадай число»:** Создайте консольную игру, в которой компьютер загадывает число, а пользователь пытается его угадать. Используйте типы данных *int* и *String*.
2. **Калькулятор с историей:** Создайте калькулятор, который хранит историю всех выполненных операций. Используйте тип данных *List<String>*.
3. **Анализатор текста:** Создайте приложение, которое анализирует текст и выводит статистику: количество слов, символов, предложений, частоту использования каждого слова. Используйте типы данных *String*, *List<String>* и *Map<String, int>*.

Дополнительные рекомендации:

1. Используйте документацию Dart для поиска информации о типах данных (<https://dart.dev/language/built-in-types>).
2. Протестируйте свои программы с различными входными данными.
3. Делайте коммиты в репозиторий после каждого завершеного задания.
4. Представить результат работы в рукописном виде в рабочей тетради и в электронном формате с использованием Git-репозитория с предоставлением QR-кода на репозиторий.