

Задание № 1: Создание функции, возвращающей сумму двух чисел.

Напишите функцию sum, которая принимает два числа и возвращает их сумму.

Описание алгоритма решения:

1. Объявите функцию sum с двумя параметрами a и b.
2. Внутри функции верните результат сложения a и b

The screenshot shows the DartPad interface. On the left, the code editor contains the following Dart code:

```
1
2
3 int sum(int a, int b) {
4     return a + b;
5 }
6
7 void main() {
8     print('Результат работы функции sum = ${sum(10,10)}');
9 }
```

On the right, the results panel displays the output of the `print` statement: "Результат работы функции sum = 20".

Рисунок №1 – Результат решения задания №1.

Задание № 2: Создание функции, возвращающей произведение двух чисел.

Напишите функцию multiply, которая принимает два числа и возвращает их произведение.

Описание алгоритма решения:

1. Объявите функцию multiply с двумя параметрами a и b.
2. Внутри функции верните результат умножения a и b.

The screenshot shows the DartPad interface. On the left, the code editor contains the following Dart code:

```
1
2
3 int multiply(int a, int b) {
4     return a * b;
5 }
6
7 void main() {
8     print('Результат работы функции multiply = ${multiply(10,10)}');
9 }
10
```

On the right, the results panel displays the output of the `print` statement: "Результат работы функции multiply = 100".

Рисунок №2 – Результат решения задания №2.

Задание № 3: Создание функции, возвращающей результат деления двух чисел

Напишите функцию *divide*, которая принимает два числа и возвращает их частное.

Описание алгоритма решения:

1. Объявите функцию *divide* с двумя параметрами *a* и *b*.
2. Внутри функции верните результат деления *a* на *b*.

The screenshot shows the DartPad interface. On the left, there is a code editor with the following Dart code:

```
1
2
3 double divide(double a, double b) {
4     return a / b;
5 }
6
7 void main() {
8     print('Результат работы функции divide = ${divide(100,10)}');
9 }
10
```

On the right, there is a results panel with the output: "Результат работы функции divide = 10".

Рисунок №3 – Результат решения задания №3.

Задание № 4: Создание функции, возвращающей остаток от деления двух чисел

Напишите функцию *remainder*, которая принимает два числа и возвращает остаток от их деления.

Описание алгоритма решения:

1. Объявите функцию *remainder* с двумя параметрами *a* и *b*.
2. Внутри функции верните результат операции *a % b*.

The screenshot shows the DartPad interface. On the left, there is a code editor with the following Dart code:

```
1
2
3 int remainder(int a, int b) {
4     return a % b;
5 }
6
7 void main() {
8     print('Результат работы функции remainder = ${remainder(13,3)}');
9 }
10
```

On the right, there is a results panel with the output: "Результат работы функции remainder = 1".

Рисунок №4 – Результат решения задания №4.

Задание № 5: Создание функции, возвращающей результат возведения числа в степень

Напишите функцию *power*, которая принимает два числа и возвращает результат возведения первого числа в степень второго.

Описание алгоритма решения:

1. Объявите функцию *power* с двумя параметрами *base* и *exponent*.
2. Внутри функции верните результат возведения *base* в степень *exponent*.

The screenshot shows a DartPad interface. On the left, the code for task 5 is displayed:

```
1 import 'dart:math';
2
3 num power(num base, num exponent) {
4   return pow(base, exponent);
5 }
6
7 void main() {
8   print('Результат работы функции power = ${power(3,3)}');
9 }
```

In the center, there are three buttons: a question mark icon, a code editor icon, and a blue "Run" button. To the right, the output window displays the result: "Результат работы функции power = 27".

Рисунок №5 – Результат решения задания №5.

Задание № 6: Создание функции, возвращающей факториал числа

Напишите функцию *factorial*, которая принимает число и возвращает его факториал.

Описание алгоритма решения:

1. Объявите функцию *factorial* с одним параметром *n*.
2. Используйте цикл *for* для вычисления факториала числа *n*.
3. Верните результат.

The screenshot shows a DartPad interface. On the left, the code for task 6 is displayed:

```
1 int factorial(int n) {
2   int result = 1;
3   for (int i = 1; i <= n; i++) {
4     result *= i;
5   }
6   return result;
7 }
8
9 void main() {
10  print('Результат работы функции factorial = ${factorial(5)}');
11 }
```

In the center, there are three buttons: a question mark icon, a code editor icon, and a blue "Run" button. To the right, the output window displays the result: "Результат работы функции factorial = 120".

Рисунок №6 – Результат решения задания №6.

Задание № 7: Создание функции, возвращающей сумму элементов списка

Напишите функцию sumList, которая принимает список чисел и возвращает их сумму.

Описание алгоритма решения:

1. Объявите функцию sumList с одним параметром *list*.
2. Используйте цикл *for* для перебора элементов списка и вычисления их суммы.
3. Верните результат.

The screenshot shows the DartPad interface. On the left, the code for `sumList` and `main` is displayed. The `sumList` function takes a list of integers and returns their sum. The `main` function prints the result of calling `sumList` with the list [1, 2, 3, 4]. On the right, the output window shows the result: "Результат работы функции sumList = 10".

```
1 int sumList(List<int> list) {  
2     int sum = 0;  
3     for (int num in list) {  
4         sum += num;  
5     }  
6     return sum;  
7 }  
8  
9 void main() {  
10    print('Результат работы функции sumList = ${sumList([1,2,3,4])}');  
11 }
```

Рисунок №7 – Результат решения задания №7.

Задание № 8: Создание функции, возвращающей среднее значение элементов списка

Напишите функцию averageList, которая принимает список чисел и возвращает их среднее значение.

Описание алгоритма решения:

1. Объявите функцию averageList с одним параметром *list*.
2. Используйте цикл *for* для перебора элементов списка и вычисления их суммы.
3. Разделите сумму на количество элементов в списке.
4. Верните результат.

The screenshot shows the DartPad interface. On the left, the code for `averageList` and `main` is displayed. The `averageList` function takes a list of integers and returns their average by dividing the sum by the length of the list. The `main` function prints the result of calling `averageList` with the list [1, 2, 3, 4]. On the right, the output window shows the result: "Результат работы функции averageList = 2.5".

```
1 double averageList(List<int> list) {  
2     int sum = 0;  
3     for (int num in list) {  
4         sum += num;  
5     }  
6     return sum / list.length;  
7 }  
8  
9 void main() {  
10    print('Результат работы функции averageList = ${averageList([1,2,3,4])}');  
11 }
```

Рисунок №8 – Результат решения задания №8.

Задание № 9: Создание функции, возвращающей максимальный элемент списка

Напишите функцию *maxElement*, которая принимает список чисел и возвращает максимальный элемент.

Описание алгоритма решения:

1. Объявите функцию *maxElement* с одним параметром *list*.
2. Используйте цикл *for* для перебора элементов списка и поиска максимального значения.
3. Верните результат.

The screenshot shows the DartPad interface. On the left, the code for the `maxElement` function is displayed:

```
1 int maxElement(List<int> list) {  
2     int max = list[0];  
3     for (int num in list) {  
4         if (num > max) {  
5             max = num;  
6         }  
7     }  
8     return max;  
9 }  
10  
11 void main() {  
12     print('Результат работы функции maxElement = ${maxElement([1,2,3,4])}');  
13 }  
14
```

On the right, the output window shows the result of running the code: "Результат работы функции maxElement = 4".

Рисунок №9 – Результат решения задания №9.

Задание № 10: Создание функции, возвращающей минимальный элемент списка

Напишите функцию *minElement*, которая принимает список чисел и возвращает минимальный элемент.

Описание алгоритма решения:

1. Объявите функцию *minElement* с одним параметром *list*.
2. Используйте цикл *for* для перебора элементов списка и поиска минимального значения.
3. Верните результат.

The screenshot shows the DartPad interface. On the left, the code for the `minElement` function is displayed:

```
1 int minElement(List<int> list) {  
2     int min = list[0];  
3     for (int num in list) {  
4         if (num < min) {  
5             min = num;  
6         }  
7     }  
8     return min;  
9 }  
10  
11 void main() {  
12     print('Результат работы функции minElement = ${minElement([1,2,3,4])}');  
13 }  
14
```

On the right, the output window shows the result of running the code: "Результат работы функции minElement = 1".

Рисунок №10 – Результат решения задания №10.

Задание № 11: Создание функции, возвращающей список квадратов чисел

Напишите функцию squareList, которая принимает список чисел и возвращает список их квадратов.

Описание алгоритма решения:

1. Объявите функцию `squareList` с одним параметром `list`.
2. Используйте цикл `for` для перебора элементов списка и вычисления их квадратов.
3. Верните новый список с квадратами.

The screenshot shows the DartPad interface. On the left, the code for the `squareList` function is displayed:

```
1 List<int> squareList(List<int> list) {  
2     List<int> squares = [];  
3     for (int num in list) {  
4         squares.add(num * num);  
5     }  
6     return squares;  
7 }  
8  
9 void main() {  
10    print('Результат работы функции squareList = ${squareList([1,2,3,4])}');  
11 }  
12
```

On the right, the results of running the code are shown:

Результат работы функции `squareList` = [1, 4, 9, 16]

Рисунок №11 – Результат решения задания №11.

Задание № 12: Создание функции, возвращающей список четных чисел

Напишите функцию evenNumbers, которая принимает список чисел и возвращает список только четных чисел.

Описание алгоритма решения:

1. Объявите функцию evenNumbers с одним параметром `list`.
2. Используйте цикл `for` для перебора элементов списка и добавления четных чисел в новый список.
3. Верните новый список.

The screenshot shows the DartPad interface. On the left, the code for the `evenNumbers` function is displayed:

```
1 List<int> evenNumbers(List<int> list) {  
2     List<int> evens = [];  
3     for (int num in list) {  
4         if (num % 2 == 0) {  
5             evens.add(num);  
6         }  
7     }  
8     return evens;  
9 }  
10  
11 void main() {  
12    print('Результат работы функции evenNumbers = ${evenNumbers([1,2,3,4])}');  
13 }  
14
```

On the right, the results of running the code are shown:

Результат работы функции `evenNumbers` = [2, 4]

Рисунок №12 – Результат решения задания №12.

Задание № 13: Создание функции, возвращающей список нечетных чисел

Напишите функцию *oddNumbers*, которая принимает список чисел и возвращает список только нечетных чисел.

Описание алгоритма решения:

1. Объявите функцию *oddNumbers* с одним параметром *list*.
2. Используйте цикл *for* для перебора элементов списка и добавления нечетных чисел в новый список.
3. Верните новый список.

The screenshot shows the DartPad interface. On the left, the code for the `oddNumbers` function is displayed:

```
1 List<int> oddNumbers(List<int> list) {  
2   List<int> odds = [];  
3   for (int num in list) {  
4     if (num % 2 != 0) {  
5       odds.add(num);  
6     }  
7   }  
8   return odds;  
9 }  
10  
11 void main() {  
12   print('Результат работы функции oddNumbers = ${oddNumbers([1,2,3,4])}');  
13 }
```

On the right, the output window shows the result of running the code: "Результат работы функции oddNumbers = [1, 3]".

Рисунок №13 – Результат решения задания №13.

Задание № 14: Создание функции, возвращающей список чисел, больших заданного

Напишите функцию *greaterThan*, которая принимает список чисел и число *threshold*, и возвращает список чисел, больших *threshold*.

Описание алгоритма решения:

1. Объявите функцию *greaterThan* с двумя параметрами *list* и *threshold*.
2. Используйте цикл *for* для перебора элементов списка и добавления чисел, больших *threshold*, в новый список.
3. Верните новый список.

The screenshot shows the DartPad interface. On the left, the code for the `greaterThan` function is displayed:

```
1 List<int> greaterThan(List<int> list, int threshold) {  
2   List<int> greater = [];  
3   for (int num in list) {  
4     if (num > threshold) {  
5       greater.add(num);  
6     }  
7   }  
8   return greater;  
9 }  
10  
11 void main() {  
12   print('Результат работы функции greaterThan = ${greaterThan([1,2,3,4],2)}');  
13 }
```

On the right, the output window shows the result of running the code: "Результат работы функции greaterThan = [3, 4]".

Рисунок №14 – Результат решения задания №14.

Задание № 15: Создание функции, возвращающей список чисел, меньших заданного

Напишите функцию *lessThan*, которая принимает список чисел и число *threshold*, и возвращает список чисел, меньших *threshold*.

Описание алгоритма решения:

1. Объявите функцию *lessThan* с двумя параметрами *list* и *threshold*.
2. Используйте цикл *for* для перебора элементов списка и добавления чисел, меньших *threshold*, в новый список.
3. Верните новый список.

The screenshot shows the DartPad interface with the following code in the editor:

```
1 List<int> lessThan(List<int> list, int threshold) {  
2     List<int> less = [];  
3     for (int num in list) {  
4         if (num < threshold) {  
5             less.add(num);  
6         }  
7     }  
8     return less;  
9 }  
10  
11 void main() {  
12     print('Результат работы функции lessThan = ${lessThan([1,2,3,4],2)}');  
13 }
```

In the results panel, it says "Результат работы функции lessThan = [1]".

Рисунок №15 – Результат решения задания №15.

Задание № 16: Создание функции, возвращающей список чисел, равных заданному

Напишите функцию *equalTo*, которая принимает список чисел и число *target*, и возвращает список чисел, равных *target*.

Описание алгоритма решения:

1. Объявите функцию *equalTo* с двумя параметрами *list* и *target*.
2. Используйте цикл *for* для перебора элементов списка и добавления чисел, равных *target*, в новый список.
3. Верните новый список.

The screenshot shows the DartPad interface with the following code in the editor:

```
1 List<int> equalTo(List<int> list, int target) {  
2     List<int> equal = [];  
3     for (int num in list) {  
4         if (num == target) {  
5             equal.add(num);  
6         }  
7     }  
8     return equal;  
9 }  
10  
11 void main() {  
12     print('Результат работы функции equalTo = ${equalTo([1,2,3,4],3)}');  
13 }
```

In the results panel, it says "Результат работы функции equalTo = [3]".

Рисунок №16 – Результат решения задания №16.

Индивидуальные задания для закрепления материала

Базовые задания:

1. **Калькулятор**: Разработайте программу, которая выполняет базовые арифметические операции (сложение, вычитание, умножение, деление) над двумя числами. Используйте функции для каждой операции.
2. **Конвертер температур**: Создайте программу, которая конвертирует температуру между градусами Цельсия, Фаренгейта и Кельвина. Используйте отдельные функции для каждого преобразования.
3. **Генератор случайных чисел**: Напишите программу, которая генерирует случайное число в заданном диапазоне. Используйте функцию rand() и реализуйте функцию для задания диапазона.
4. **Проверка на простоту**: Разработайте функцию, которая проверяет, является ли заданное число простым. Используйте эту функцию в программе, которая находит все простые числа в заданном диапазоне.
5. **Сортировка массива**: Напишите программу, которая сортирует массив чисел по возрастанию. Используйте функцию для реализации алгоритма сортировки (например, пузырьковая сортировка).
6. **Поиск максимального элемента**: Разработайте функцию, которая находит максимальный элемент в массиве. Используйте эту функцию в программе, которая находит максимальный элемент в двумерном массиве.
7. **Работа со строками**: Напишите программу, которая подсчитывает количество символов, слов и строк в тексте. Используйте функции для каждой операции.
8. **Калькулятор площади фигур**: Разработайте программу, которая вычисляет площадь различных геометрических фигур (круг, прямоугольник, треугольник). Используйте отдельные функции для каждой фигуры.
9. **Игра «Угадай число»**: Напишите программу, в которой компьютер загадывает число, а пользователь пытается его угадать. Используйте функции для генерации случайного числа и проверки предположения пользователя.

Средний уровень:

1. **Система управления задачами**: Создайте консольное приложение для управления задачами. Пользователь может добавлять, просматривать, редактировать и удалять задачи. Используйте функции для каждого действия.
2. **Телефонный справочник**: Разработайте программу для хранения и управления телефонным справочником. Пользователь может добавлять, искать, редактировать и удалять записи. Используйте функции для каждой операции.
3. **Программа для шифрования текста**: Напишите программу, которая шифрует и дешифрует текст с использованием шифра Цезаря. Используйте функции для шифрования и дешифрования.
4. **Система управления банковским счетом**: Разработайте программу, которая позволяет пользователю управлять банковским счетом (пополнение, снятие, просмотр баланса). Используйте функции для каждой операции.
5. **Игра «Сапер»**: Напишите программу, реализующую игру "Сапер". Используйте функции для генерации поля, отображения поля, обработки хода и проверки окончания игры.
6. **Программа для работы с матрицами**: Разработайте программу, которая выполняет основные операции с матрицами (сложение, вычитание, умножение). Используйте функции для каждой операции.
7. **Система управления библиотекой**: Создайте программу для управления библиотекой. Пользователь может добавлять, искать, редактировать и удалять книги. Используйте функции для каждой операции.
8. **Программа для решения квадратных уравнений**: Напишите программу, которая решает квадратные уравнения. Используйте функции для вычисления дискriminанта и корней уравнения.

Продвинутый уровень:

1. **Программа для работы с графами**: Разработайте программу, которая позволяет пользователю создавать, редактировать и анализировать графы. Используйте функции для добавления и удаления вершин и ребер, поиска кратчайшего пути и т.д.
2. **Программа для работы с большими числами**: Разработайте программу, которая выполняет арифметические операции с большими числами (больше, чем может хранить стандартный тип данных). Используйте функции для сложения, вычитания, умножения и деления.
3. **Программа для работы с регулярными выражениями**: Напишите программу, которая позволяет пользователю работать с регулярными выражениями (поиск, замена, проверка соответствия). Используйте функции для каждой операции.