Первая программа

// BEGIN (write your solution here)

console.log('Hello, World!');

// END

Математика и константы

// BEGIN (write your solution here)

const weight = 72; // My weight

const c = 300000000; // c of the light

const energy = weight \* (c\*\*2);

export default energy;

console.info(`Your body energy is ${energy} joules. This is 1 / ${Math.round(389000000000000000000 / energy)} of the yearly world energy consumption. Woah!`);

// END

Функции и ящики

const squareOfSum = (a, b) => (a\*a) + (2 \* a \* b) + (b\*b);

export default squareOfSum;

Условия и принятия решений

// BEGIN (write your solution here)

const finalGrade = (exam, projects) => {

if (exam > 90 || projects > 10) {

return 100;

} else if (exam > 75 || projects > 5) {

return 90;

} else if (exam > 50 || projects > 2) {

return 75;

} else {

return 0;

}

};

// END

export default finalGrade;

if (a > 100) {

something;

} else if (a === 95) {

something\_different;

} else if (a === 90) {

something\_different\_yet;

} else {

something\_else;

// BEGIN (write your solution here)

const finalGrade = (exam, projects) => {

if (exam > 90 || projects > 10) {

return 100;

}

if (exam > 75 && projects >= 5) {

return 90;

}

if (exam > 50 && projects >= 2) {

return 75;

}

return 0;

};

// END

export default finalGrade;

Ошибки и как с ними справляться

Reference.js

const reference = () => {

// BEGIN (write your solution here)

function hoist() {

  a = 20;

  var b = 100;

}

hoist();

console.log(b);

  // END

};

export default reference;

type.js

const type = () => {

  // BEGIN (write your solution here)

var foo = undefined;

foo();

  // END

};

export default type;

Рекурсия

const sequenceSum = (begin, end) => {

  // BEGIN (write your solution here)

    if (begin > end) {

      return NaN;

     }

    if (begin === end) {

      return begin;

    }

      return begin + sequenceSum(begin + 1, end);

  // END

};

export default sequenceSum;

Интеративный процесс

const smallestDivisor = (num) => {

  // BEGIN (write your solution here)

    const iter = (acc) => {

    // We use 'num / 2' in the condition below, and not 'num'.

    // This is a simple optimization: a number cannot be divided

    // by a number larger than its half.

    if (acc > num / 2) {

      return num;

    }

    if (num % acc === 0) {

      return acc;

    }

    return iter(acc + 1);

  };

  return iter(2);

  // END

};

Источник <https://gist.github.com/Lodo4ka/8f0cb3cff39ee45130172be3bfb96642>

Переменные, декларативное императивное программирование

const smallestDivisor = (num) => {

// BEGIN (write your solution here)

 if (num < 1) {

    return NaN;

  }

  if (num === 1) {

    return num;

  }

  let divisor = 2;

  while (num % divisor !== 0) {

    divisor += 1;

  }

  return divisor;

// END

};

export default smallestDivisor;

Строки и работа с символами

// BEGIN (write your solution here)

 const reverse = (str) => {

   let result = '';

  for (let i = str.length - 1; i >= 0 ; ){

    result = result + str[i];

    i = i - 1;

  }

  return result;

}

// END

export default reverse;

Цикл For и изменение переменных

// BEGIN (write your solution here)

const isPrime = (num) => {

  if (num < 2) {

  return false;

  }

  for (let i = 2; i < num; i += 1)

    if (num % i === 0) {

    return false;

  }

  return true;

};

// END

export default isPrime;

Тесты и автоматическая проверка кода

import assert from 'assert';

import factorial from './implementations';

// Example

assert.equal(factorial(2), 2);

assert.equal(factorial(3), 6);

// BEGIN (write your solution here)

assert.equal(factorial(0), 1);

assert.equal(factorial(1), 1);

assert.equal(factorial(8), 40320);

// END

Модули

 файле myMathModule.js:

export const getTrianglePerimeter = (a, b, c) => a + b + c;

// BEGIN (write your solution here)

export const getTriangleArea = (h, b) => {

  const area = (h \* b) / 2;

  return area;

};

// END

В файле solution.js:

import { square } from './square';

import { getTriangleArea } from './myMathModule';

// BEGIN (write your solution here)

const solution = (n) => getTriangleArea(n, square(n) / 2);

export default solution;

// END

Типизация

import { length } from './strings'; // eslint-disable-line

// BEGIN (write your solution here)

const addDigits = (num) => {

  num += '';

  let sum = 0;

  for(let i = 0; i < num.length; i++) {

     sum += Number(num[i]);

  }

  if(num.length > 1) {

    return addDigits(sum);

  }

  return sum;

}

export default addDigits;

// END

Чистые функции

import { length, toUpperCase } from './strings';

// BEGIN

const solution = (str) => {

  let result = '';

  for (let i = 0; i < length(str); i += 1) {

    if (str[i] !== ' ' && (str[i - 1] === ' ' || i === 0 )) {

      result += toUpperCase(str[i]);

    } else {

      result += str[i];

    }

  }

  return result;

};

export default solution;

// END