

情報の授業

アルゴリズムとプログラム Ver. 4.2

TPOINT

学習教材

半角の数字 1, 2, 3
漢字(全角)の数字 一, 二, 三
Scratchでは、漢字の数字を入れると正しく動作しません。
今回の実習中は、漢字(日本語)の入力モードを切って(オフ)使ってください。



見た目いっしょだけど、右側は漢字の「一」なので、0として扱われ正しく動かない

次見て



Go.Ota, Oct.2021 1

プログラムに名前をつけて保存 準備運動1

理解

TPOINT



プログラムの名前を指定
この場合は例えば、
Junbi1

新しく作る場合は**新規**

これからの課題や打ち込むプログラムは前に作ったものを改造すると楽なものが多くなります。個々のプログラムにつけて保存したり、コピーを保存で流用したりしましょう。

今作っているプログラムはネット上に自動保存されます。確実に保存したい人は**直ちに保存**を指定します。

2

準備運動1: Scratchの四則演算

理解・打込み



そのまま打ち込む
プログラム

プログラムが出来たら旗を押して動かしてみよう

TPOINT

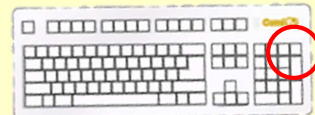
次見て

まず、準備運動で、四則演算(足し算、引き算、掛け算、割り算)のプログラムを作って動かしてみます。

- ・数値は好きな値をいれてください。
- ・普通の数学の記号が、プログラムでは違うものがあります。

掛け算 *
割り算 /

(キーボードの右側にテンキーがついている時はこれらの記号があります)



3

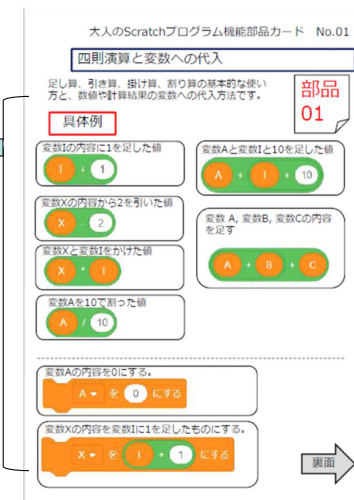
準備運動1: Scratchの四則演算

理解・打込み



部品
01

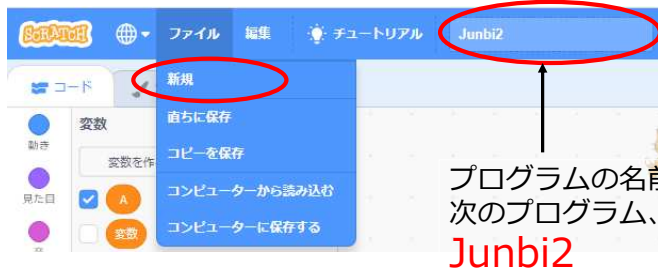
示された部品カードに、使える似たようなパターンがあるので、それを参考にして作成する。(そのままの、答えがあるわけではない)



4

新しいプログラムを作る 準備運動2

理解



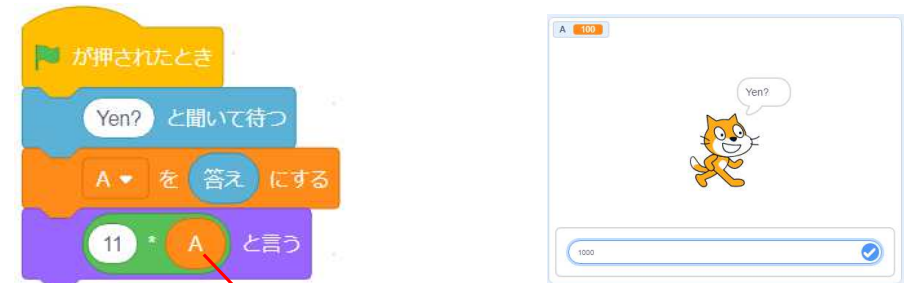
プログラムの名前を指定
次のプログラム、
Junbi2

新しいプログラムは
新規で作成

5

準備運動2: 円から韓国ウォンに変換。

理解・打込み



そのまま打ち込む
プログラム

変数の **A** を使う

日本円の金額を入れて、韓国のウォンがいくらになるか計算します。
1円=11ウォンで計算しています。

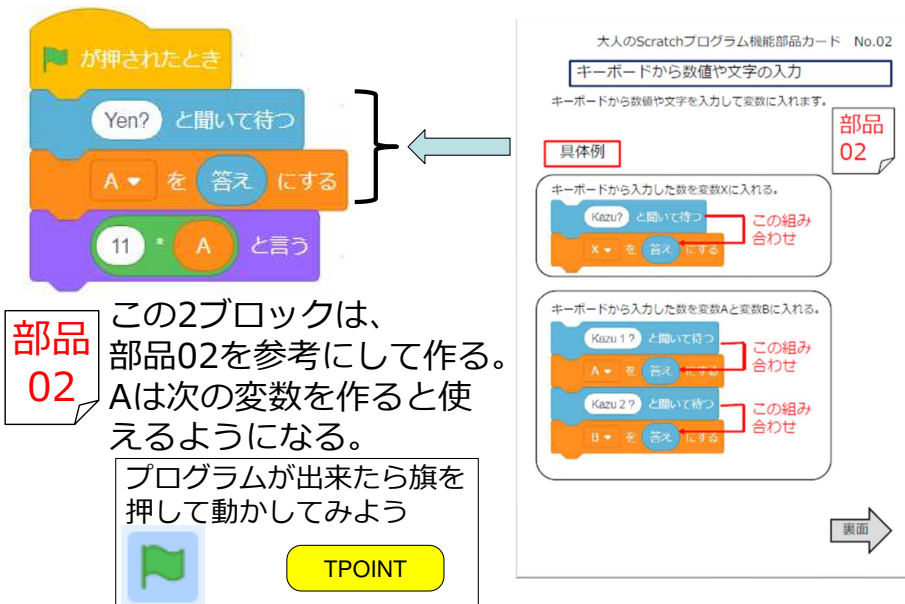
次のスライド
のヒント
も見て

次見て

6

準備運動2: 円から韓国ウォンに変換。

理解・打込み



部品 02 この2ブロックは、
部品02を参考にして作る。
Aは次の変数を作ると使
えるようになる。

プログラムが出来たら旗を
押して動かしてみよう

TPOINT

次見て

7

プログラムと変数

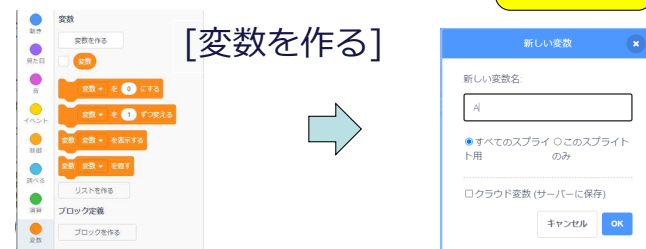


変数は数が入る箱みたいなものです。

Aという名前の変数を作って使います。

TPOINT

[変数を作る]



次見て

8

プログラムを「作る」と「動かす」を区別しよう

プログラムを作る。



キーボードから
プログラムの中
の数値を入力。



又は、プログラムが最後まで実行

プログラムを動かす。



入力した数字が入る。

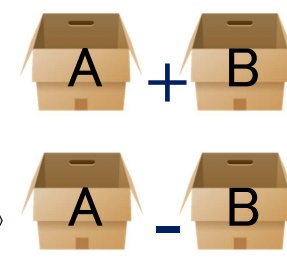
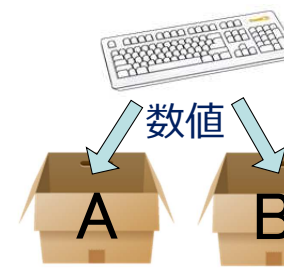
キーボードから
データを入力。



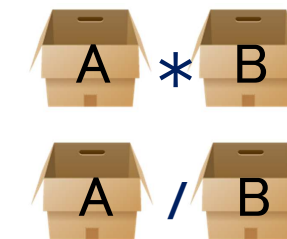
9

課題1: 入力した2個の数で四則演算

開発



課題1:
2個の数をキーボードから入力して、
その 足し算、引き算、掛け算、割
り算の結果を順番に表示します。



部品
01

部品
02

次見て

10

課題1: 入力した2個の数で四則演算

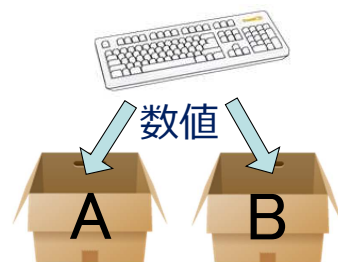
ヒント



足し算、引き算、掛け算、
割り算は準備運動1でやっ
ています。

直接数字を指定して入れけ
ど、これをAとBの変数に
する。

次見て



部品
02

変数Aと変数Bに入力する
方法は、部品カード02を
見てください。

11

課題1: 入力した2個の数で四則演算

TPOINT

ヒント



AとBの変数に数をキー
ボードから入力する。



計算はすべて、AとBの変
数で行う。

プログラムが出来たら旗を
押して動かしてみよう



大人のScratchプログラム機能部品カード No.02
キーボードから数値や文字の入力
キーボードから数値や文字を入力して変数に入れます。

具体例



部品
02

部品
02

12

打ち込み1: 合格判断

理解・打ち込み



そのまま打ち込む
プログラム

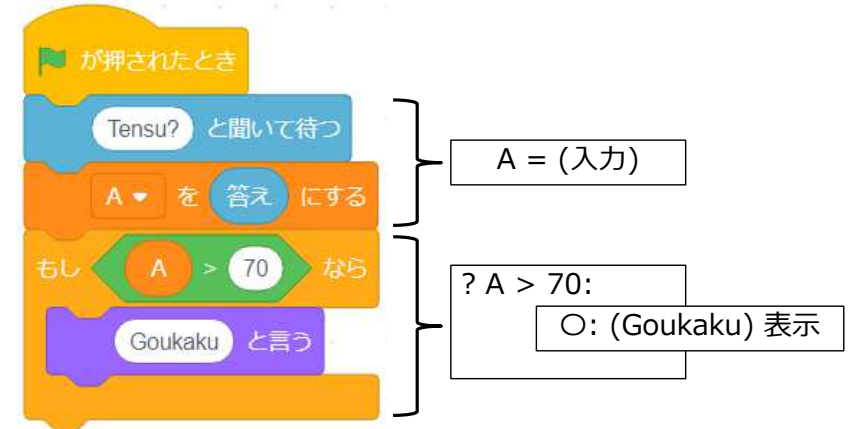
変数Aに数を入力して、70より大きければ(70は入らない)、ネコにGoukaku(合格)と言わせてみます。

次見て

13

打ち込み1: 合格判断の図式

理解



部品
03

部品カード03も見てく
ださい。

プログラムが出来たら旗を
押して動かしてみよう

14

課題2: 合格不合格判断

開発

変数Aに数を入力して、
もし、70より大きい**なら**(70は入らない)、ネコにGoukaku(合格)と言わせます。
そうで**なければ**、ネコにFugoukaku(不合格)と言わせます。

部品
04

TPOINT

ヒント: 部品04を参考にして作ってみてください。
部品の内容そのままでは使えません。



15

課題3: 成績A～C

開発

変数Aに数を入力して、**もし**、70より大きい**なら**(70は入らない)、Seiseki A(成績A), 50より大きい**なら**(50は入らない)、Seiseki B(成績B), それ以外(50以下だったら)、Seiseki C(成績C), とネコに言わせます。

ヒント: 課題2を流用して
(コピーして保存)して作る。

部品
05

ヒント: 部品05を参考にして作ってみてください。



16

打ち込み2: 1から10までの数言う

理解・打ち込み

部品 06

そのまま打ち込むプログラム

が押されたとき

A を 0 にする

A = 10 まで繰り返す

A を A + 1 にする

A と 1 秒言う

1,2,3,...,10までの数をネコと順番にと言わせてみます。

次見て

17

プログラムと図式

理解

プログラムもだんだん複雑になってきます。プログラムがどのような部品でくみあがっているか分かりやすくするため、今後ヒントで図式を示します。

部品 06

A = 1,2,...
I = 10まで

Aを表示

A と 2 秒言う

が押されたとき

A を 0 にする

A = 10 まで繰り返す

A を A + 1 にする

A と 1 秒言う

次見て

18

打ち込み3: 1から10までの合計

理解・打ち込み

部品 06

そのまま打ち込むプログラム

が押されたとき

S を 0 にする

A を 0 にする

A = 10 まで繰り返す

A を A + 1 にする

A と 1 秒言う

S を S + A にする

S と言う

1,2,3,...,10までの数をネコが順番に言いながら、その数を足して、最後にその合計を言います。

ヒント: 打ち込み2を流用して(コピーして保存)して作る。

次見て

19

プログラムの図式

TPOINT

理解

「1から10までの合計」は次のような図式になります。

S = 0

A = 1,2,...
I = 10まで

Aを表示

S = S + A

Sを表示

図式でもヒントがでますので、プログラムの図式の対応を考えて

部品 06

部品6を見てね。

が押されたとき

S を 0 にする

A を 0 にする

A = 10 まで繰り返す

A を A + 1 にする

A と 1 秒言う

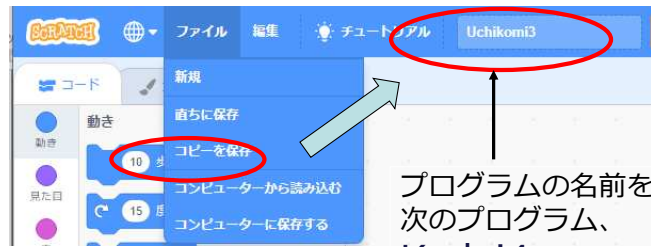
S を S + A にする

S と言う

20

他のプログラムを流用して作る

理解



プログラムの名前を指定
次のプログラム、
Kadai4

○手順1

今回は、「Uchikomi3」を利用して、新しいプログラム
「Kadai4」を新規につくる場合は

コピーを保存で作成

(「打ち込み3」は残って、新しく打ち込み3と同じものが
残る)

○手順2

ファイルの名前を「課題4」に変える。

21

課題4: 2からXまでの偶数の合計

開発

Xの値を入力して、2からX以下の偶数の合
計を求めるプログラムを作成してください。
Xには偶数だけ入力します。

TPOINT

例: X = 6の場合、2+4+6

X = 10の場合、2+4+6+8+10

重要! 下記の順番で作って!!

打ち込み3: 「1から10までの合計」のプログラム
を流用してして作る。

手順1: 次のスライド

まず、2から10までの偶数の合計を求めるプログ
ラムを作ります。一か所だけ変更

手順2: 次の次のスライド

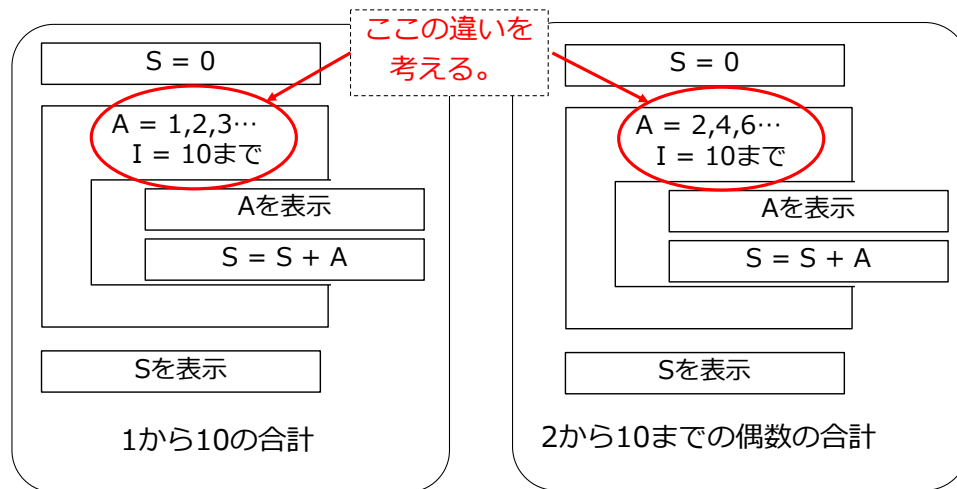
次に、Xを入力して、X以下の合計を求めるプログ
ラムを作ると楽かもしれません。

次見て

22

課題4 ヒント2: 2から10以下の偶数の合計

ヒント



部品
06

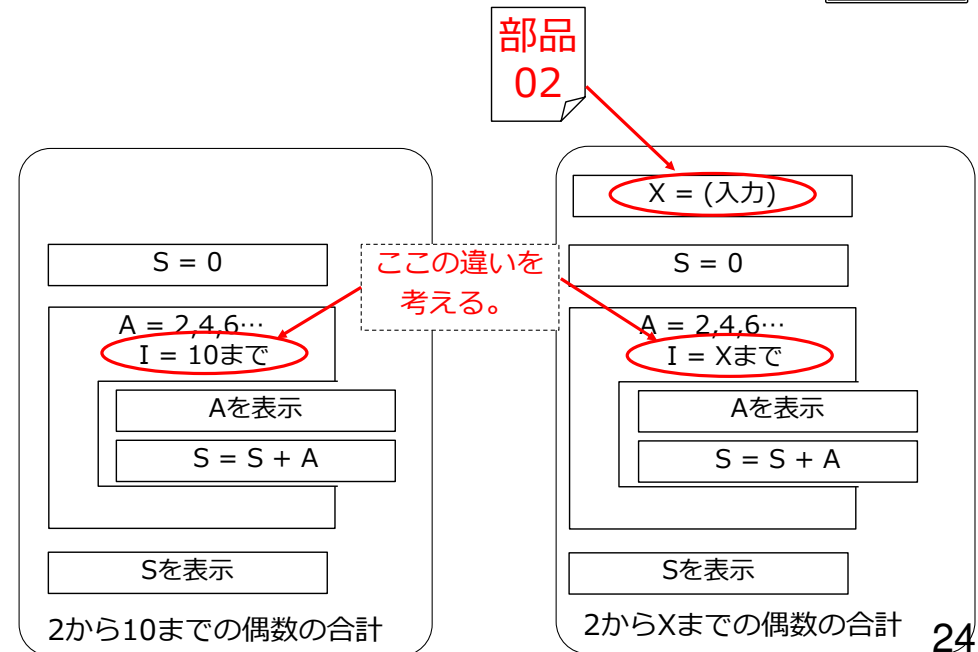
2, 4, 6となるように部品06を少し
変更して使います。

次見て

23

課題4補足: 2からX以下の偶数の合計

ヒント



24

たくさんの数の合計の説明

次からたくさん数を入力して合計を求めるプログラムをつくっていきます。



今まで、数を入力するのに変数をつくっていきました。課題3では、A,S,Xの3つの変数を使って合計を計算しました。ただし、が、たくさん数を入力するには、A,B,C,D,E・・・と多くの変数を使うのは大変です。

こんな時にリスト(配列)を使うと便利です。次からリスト(配列)を学習していきます。

25

打ち込み4: おみくじをつくる

理解・打ち込み

リストを使っておみくじを作ってみます。



そのまま打ち込むプログラム

ネコをクリックするとDai-Kichi(大吉), Chu-Kichi(中吉), Syo-Kichi(小吉), Kyou(凶), Dai-Kyou(大凶)とランダムに言います。

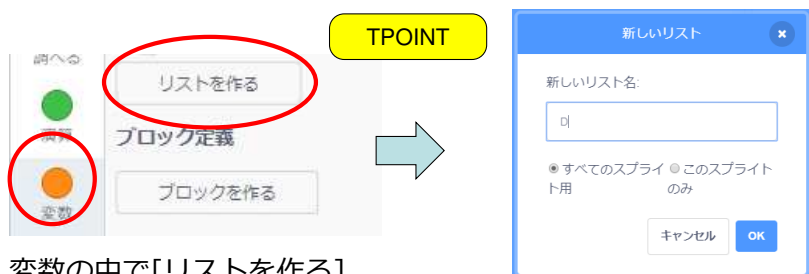
部品
07

次見て

26

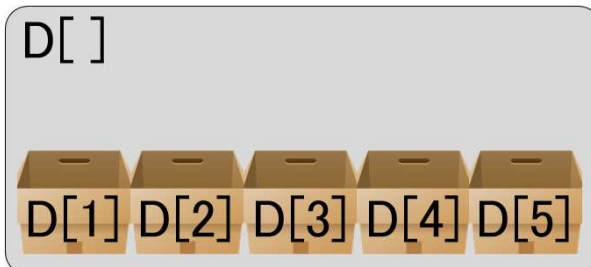
打ち込み4: 配列(リスト)を作る

ヒント



変数の中で[リストを作る]を指定。

リスト名を指定して作成

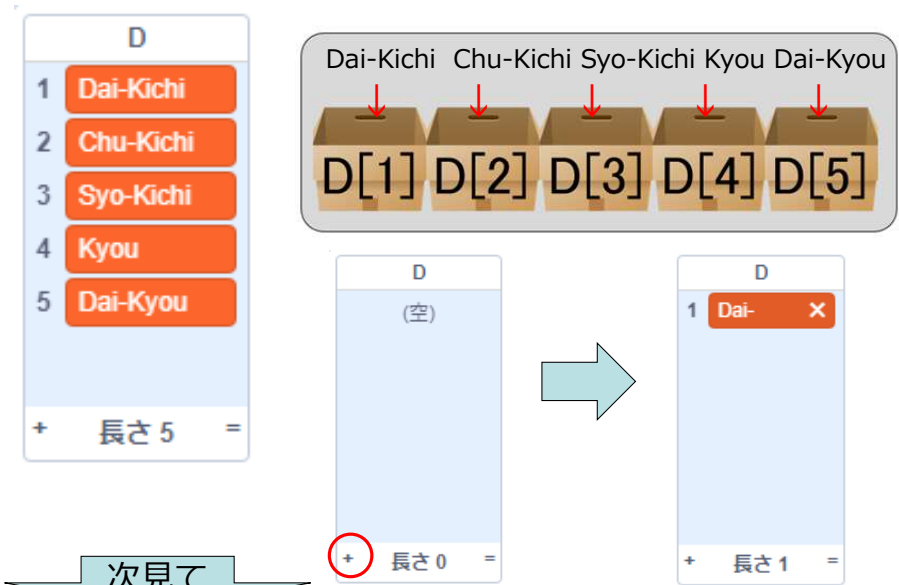


次見て

リストはDに番号がついている、さくたんの箱のイメージです。 27

打ち込み4: 配列(リスト)に値を入れる

ヒント



+ を押す

入力する

28

打ち込み4: プログラムの動作

ヒント

このスプライトが押されたとき

I を 1 から 5 までの乱数 にする

D の I 番目 と言う

変数I(アイ)に1~5の
どれかの数が入る。

ヒント: 変数 I(アイ)を
作って使います

D[I(アイ)]番目 の内容を言
う(Iに入っている数に対応し
たD[]の内容)



29

打ち込み5: リストを使った5つの数の合計

理解・打ち込み

D

1	6
2	10
3	8
4	3
5	6

+ 長さ 5 =

が押されたとき

S を 0 にする

I を 0 にする

I = 5 まで繰り返す

I を I + 1 にする

S を S + D の I 番目 にする

S と言う

そのまま打ち込む
プログラム

いきなりプログラムが難しくなりまし
た。あらかじめリストDに入れておい
た5個の数の合計を計算します。

次見て

30

打ち込み5補足: リストを使った5つの数の合計

理解

S = 0

I = 1, 2, 3...
I = 5 まで

Iを表示

S = S + D[I]

Sを表示

S を 0 にする

I を 0 にする

I = 5 まで繰り返す

I を I + 1 にする

I と 1 秒言う

S を S + D の I 番目 にする

S と言う

リストDに5個の数を変数にSに加えて行って合計を計算

部品
07

部品
09

31

最終目標の課題の説明: 数の並び替え

理解

予めリストD[1]からD[5]まで数を入れてお
きます。

この中の数を小さい順番に並び替えてリスト
D[1]からD[5]に入れなおしてください。

これが最後の目標の課題です。できると思う人は、い
きなり開発してもいいです。
少し難しいと思う人は、次の課題を順番にやっていき
ましょう。

- ・配列から数を見つける
- ・配列の中の一番小さい数を見つける
- ・配列の中の一番小さい数を、配列の先頭に入れる。
- ・二重のループを使う
- ・並び替えの開発

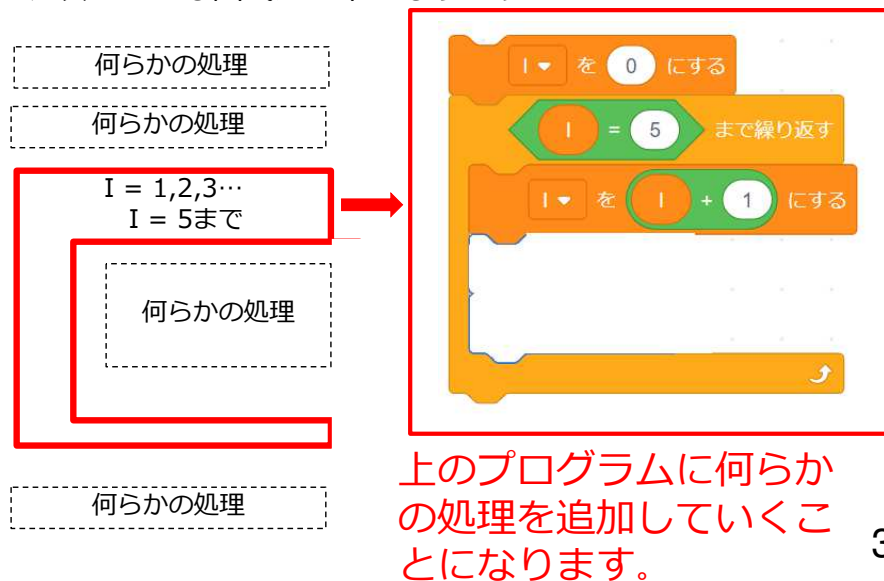


32

非常に重要

TPOINT

課題5～7までは、5個の箱のリストを順番に処理するため、次のような図式が基本になります。



33

課題5: リストの中から数を探す(検索)

課題

検索というこんな動作をするプログラムを作ります



初めにリストに数を入れておきます

使用禁止!!!!!!

次のブロックの使用は禁止します。

D の 中の なにか の場所

次に探す数を決めます
例えば 39

探す数が何番目に入っているか求めます。
この場合は、 4

34

課題5: リストの中から数を探す(検索)

ヒント1

プログラムの中で次のように変数使います



初めにリストに数を入れておきます

A = (入力)

次に探す数をキーボードから入力して変数Aに入れます。

A 39

プログラムで、数の入っている場所を探して変数Xに入れます。

X 4

次見て

35

課題5補足: 配列の中から数を探す

ヒント2

プログラムの中で数の探し方の考え方



A 39

変数Iを1～5まで変えていき、D[I]とAの内容を比較します。

D の I 番目

もし等しければ、I(何番目かの番号)の値をXに入れます。

X 4

次見て

36

課題5補足: リストの中から数を探す(検索)

ヒント3

D[]に5個の数をいれとく

A = (入力)

X = 0

I = 1, 2, 3...
I = 5まで

変数AとD[I]の内容が
同じだったら、その
時のIの値を変数Xに
入れる

Xを表示

部品
02

この部分を考えて、プログラムを
完成されてください。

部品
03

部品
07

37

課題6: リストの中の一番小さい数を見つける

こんな動作をするプログラムを作ります



初めにリス
トに数を入
れておきま
す

プログラムで、一番小
さな数を探します。

X 8

プログラムが終わった時
点で、一番小さな数が変
数Xに入っています。

次見て

38

課題6: リストの中の一番小さい数を見つける

ヒント1



X 16

変数Iを1~5まで変えて
いき、変えるときのD[I]
とXの内容を比較します。

D の I 番目

変数XとD[I]の内容を比較してD[I]が小
さければ、その値を変数Xに入れる

X 16

39

課題6補足: リストの中の一番小さい数を見つける

ヒント2

D[]に5個の数をいれとく

X = D[1]

ポイント:

変数Xに初め0が入っていると他の数
より小さいので初めにD[1]の値を入
れています。

I = 1, 2, 3...
I = 5まで

この部分を考えて、プログラムを
完成されてください。

変数XとD[I]の内容を
比較して、D[I]が小
さければ、その時の
値を変数Xに入れる

部品
03

部品
07

Xを表示

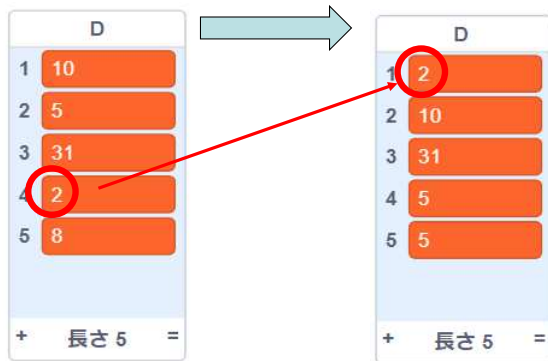
部品
09

40

課題7: 一番小さい数を配列の先頭に入れ替える

こんな動作をするプログラムを作ります

開発



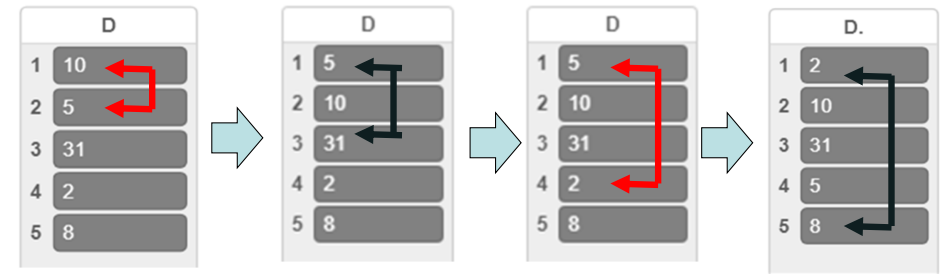
一番小さい数がD[1]に入ります。但し、リストには元のリストにあったすべての数が残っています。

次見て

41

課題7補足: 一番小さい数を配列の先頭に入れ替えるプログラムの考え方

ヒント1



1番目と2番目を比較して5の方が小さいので入れ替える

1番目と3番目を比較して5の方が小さいので入れ替えない

1番目と4番目を比較して2の方が小さいので入れ替え

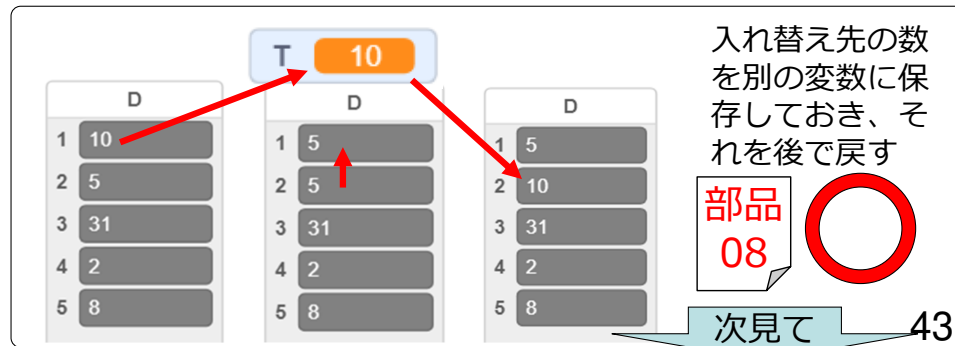
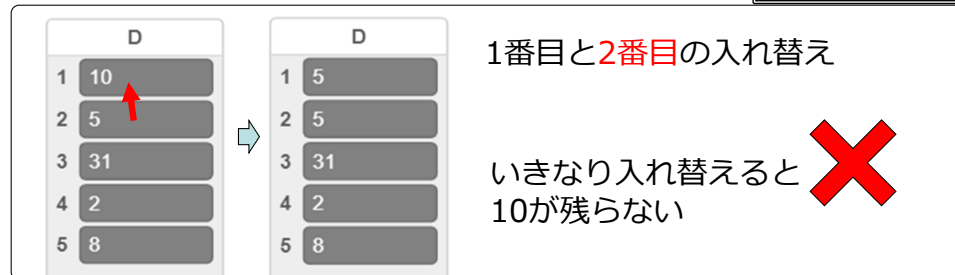
1番目と5番目を比較して2の方が小さいので入れ替えない

次見て

42

課題7補足: 一番小さい数を配列の先頭に入れ替える数の入れ替え

ヒント2



43

課題7補足: 一番小さい数を配列の先頭に入れ替える

ヒント3

D[]に5個の数をいれとく

I = 1, 2, 3...
I = 5まで

変数D[1]とD[I]の内容を比較して、D[I]が小さければ、D[1]とD[I]を入れ替える

この部分を考えて、プログラムを完成させてください。

部品 03

部品 08

部品 08

部品 09

44

打ち込み6:二重繰り返しで九九に挑戦 理解・打ち込み

二重繰り返しを使って、九九を言うプログラムを作ってみましょう



1, 2, 3, 4, 5, 6, 7, 8, 9
2, 4, 6, 8, 10, 12, 14, 16, 18
3, 6, 9, 12, 15, 18, 21, 24, 27
.....
9, 18, 27, 36, 45, 54, 63, 72, 81

1 x 1, 1 x 2, 1 x 3, 1 x 4, 1 x 5, 1 x 6, 1 x 7, 1 x 8, 1 x 9
2 x 1, 2 x 2, 2 x 3, 2 x 4, 2 x 5, 2 x 6, 2 x 7, 2 x 8, 2 x 9
3 x 1, 3 x 2, 3 x 3, 3 x 4, 3 x 5, 3 x 6, 3 x 7, 3 x 8, 3 x 9
.....
9 x 1, 9 x 2, 9 x 3, 9 x 4, 9 x 5, 9 x 6, 9 x 7, 9 x 8, 9 x 9

次見て

45

打ち込み:二重繰り返しに挑戦

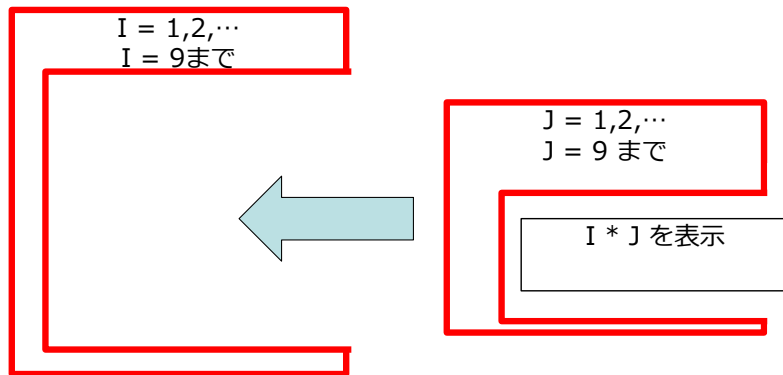
ヒント



46

非常に重要

TPOINT



部品 10

今まで使っていた、変数Iを使った繰り返しの中に、さらに繰り返しが入る形式
最後の課題の並び替えまで、これが基本的な構造になります。

47

コラム1:

TPOINT:早い段階に説明の方が良い

気づきにくい間違い:

変数名のつもりで文字にしていた



足し算のつもりで掛け算になっていた



「にする」が、「ずつ変える」にしていた



「まで繰り返す」が、「回繰り返す」にしていた



48

打ち込み7:複雑な二重繰り返しに挑戦 理解・打ち込み

二重繰り返しですが、内側の繰り返しの始まりの数を変えます。
 予めリストD[1]からD[5]まで数を入れておきます。
 初めにD[1]からD[5]までの数
 次にD[2]からD[5]までの数

 最期にD[5]までの数を言うプログラムを作ります。

例えば、D[1]~[5]までに、
 12, 9, 65, 87, 44を入れておいた時



12, 9, 65, 87, 44
 9, 65, 87, 44
 65, 87, 44
 87, 44
 44



部品
11

次のスライドにプログラムと図式。
 次の次のスライドに解説があります。

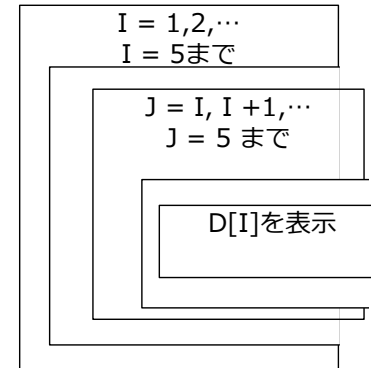
次見て

49

打ち込み:複雑な二重繰り返しに挑戦

ヒント1

D[]に5個の数をいれとく



部品
09

部品
11

次見て



I - 1に
 するのが
 ポイント

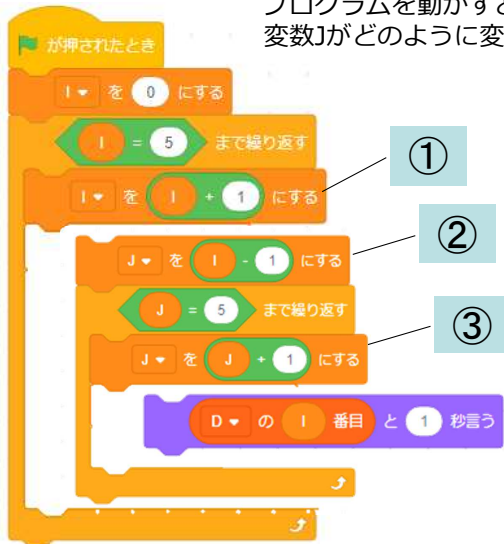
そのまま打ち込む
 プログラム(間はあけない)

50

複雑な二重繰り返しに挑戦

ヒント2

プログラムを動かすと①②③のどこを通過して、変数Iと
 変数Jがどのように変わるか考えてみましょう。



通る場所	I	J
①	1	?
②	1	0
③	1	1
③	1	2
③	1	3
③	1	4
③	1	5
①	2	5
②	2	1
③	2	2
③	2	3
③	2	4
続いています		

51

課題8:数の並び替え

開発

予めリストD[1]からD[5]まで数を入れておきます。この中の数を小さい順番に並び替えてリストD[1]からD[5]に入れなおしてください。

D	
1	10
2	5
3	31
4	2
5	8



D	
1	2
2	5
3	8
4	10
5	31



プログラムを実行した後は、数が大きくなるように入れ替えが行われています。

次見て

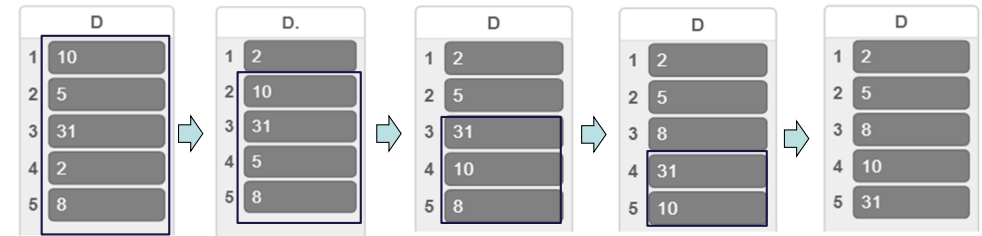
53



最期のチャレンジ:数の並び替え

ヒント1

考え方「一番小さい数をリストの先頭に入れ替える」を繰り返し実施



1回目
D[1]からD[5]で一番小さい数を配列の先頭D[1]に入れ替える

2回目
D[1]に一番小さい数はいっているので、D[2]からD[5]で一番小さい数を配列のD[2]に入れ替える

3回目
D[3]からD[5]で一番小さい数を配列のD[3]に入れ替える

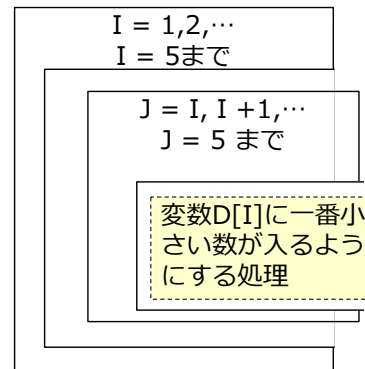
4回目
D[4]からD[5]で一番小さい数を配列のD[4]に入れ替える

54

最期のチャレンジ:数の並び替え

ヒント2

D[]に5個の数をいれとく



ヒント:

打ち込み7:複雑な二重繰り返しの挑戦のプログラムを流用して(コピーして保存)して作る。

この部分を考えて、プログラムを完成させてください。課題7のこのあたりのプログラムを少し変えるだけでできます。

部品
03

部品
08

部品
08

部品
09

部品
11

発展課題1: FizzBuss

開発

Fizz Buzzという遊びのプログラムを作ってみましょう。例えば、1から30までの数を順番に言いますが。

3の倍数の時は"Fizz"と言います。
5の場合の時は"Buzz"と言います。
3と5の倍数の時は、"Fizz Buzz"と言います。
それ以外は数字と言います。

考え方

- ・まず1から30までの数をつくる構造を考えます。
- ・その中で、3の倍数かどうか判断します。3の倍数で無ければ、5の場合数が判断します。5の倍数の場合は、"Buss"といます。3の場合数でも5の倍数でもなければ数字をいいます。
- ・3の倍数だった場合は、5の倍数が判断して、5の倍数でもあったら"Fizz Buzz"と言います。5の倍数でなかったら3の倍数なので"Fizz"と言います。

56

発展課題2: バブルソート

開発

今回作成したプログラムは選択ソートという方法を使ったもので、データを並び替える方法はいろいろあります。

次の並び替え方法の考え方やプログラムをWebで調べて作成してください。

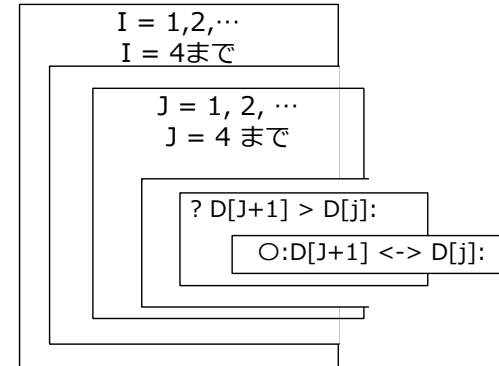
バブルソート

次のヒントの
スライドも見て

57

バブルソートの図式

D[]に5個の数をいれとく



小さいものをD[]
の番号の小さいもの
にする処理

部品
08

部品
09

部品
10

発展課題2: 並び替えの方法の処理速度の違い

開発

最期の課題です。

データの並び替えに、いろいろな方法があるのは、用途や処理速度に違いがあるからです。

今回開発した
選択ソート/バブルソートで処理速度がどのように違うか測定してみましょう。

タイマーをリセット

ここに並び替えをするプログラムを入れます

A を タイマー にする

Scratchでタイマーを使うと処理にかかった時間が測定できます。

次のヒントの
スライドも見て

59

配列に数をセットする方法

ヒント

ランダムな数を設定する。たくさん
の数を設定する時に良い

1から100の目が出るサイコロを振って、順番に数値を入れているイメージです。



部品
09

60