

2022.07.02

日本情報科教育学会 第15回全国大会

# 高等学校「情報Ⅰ」における 整列アルゴリズムの可視化プログラミング

稲川 孝司

帝塚山学院大学

喜家村 奨

帝塚山学院大学

西野 和典

太成学院大学

高橋 参吉

NPO法人

学習開発研究所

2020年度JSPS科研費JP20K02528助成による研究

# 情報 I の単元

(1) 情報社会の問題解決

(2) コミュニケーションと情報デザイン

(3) コンピュータとプログラミング

(4) 情報通信ネットワークとデータの活用

# 情報 I (3)の内容と学習の仕方

## アルゴリズムの表現

- ・フローチャート
- ・アクティビティ図

## 典型的な例

- ・並べ替え(ソート)
- ・探索(サーチ)

## 問題の発見・解決に応じたもの

- ・音声の認識と応答
- ・計測・制御
- ・画像処理
- ・物理シミュレーション
- ・自然界のシミュレーション

## プログラムを学ぶ

- ・プログラムの有用性
- ・アルゴリズムによる効率の違い

## 関数の使用による構造化

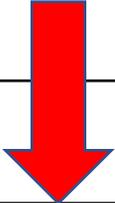
## プログラムで学ぶ

- ・形や色
- ・コマンドの仕組み

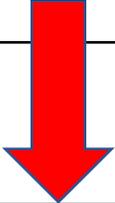
※短いプログラムでコンピュータの仕組みを学習

# 並べ替えプログラミング学習の問題点と対策

**問題点**：プログラムが正しく動作しないときに、プログラムがどのように動作しているかわからず、修正できない。



**原因**：プログラムの誤った動作の元となる場所や動作が特定できない



**対策**：変数や配列の値や動作を可視化する

# micro:bit + 拡張ディスプレイの利用

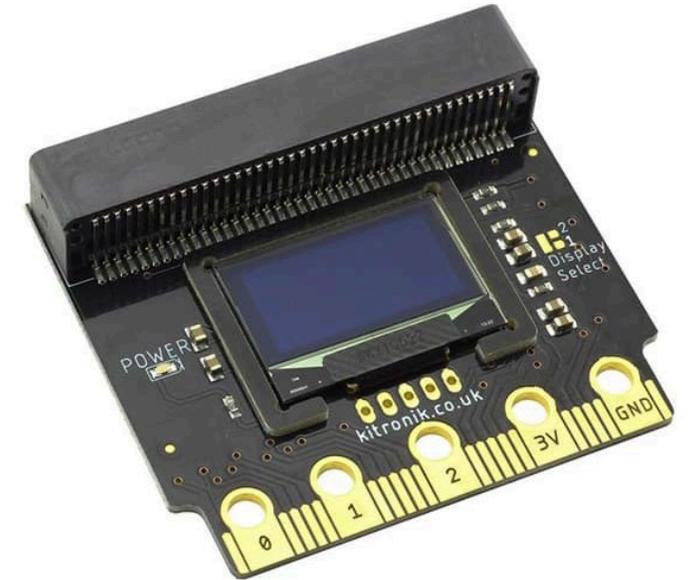
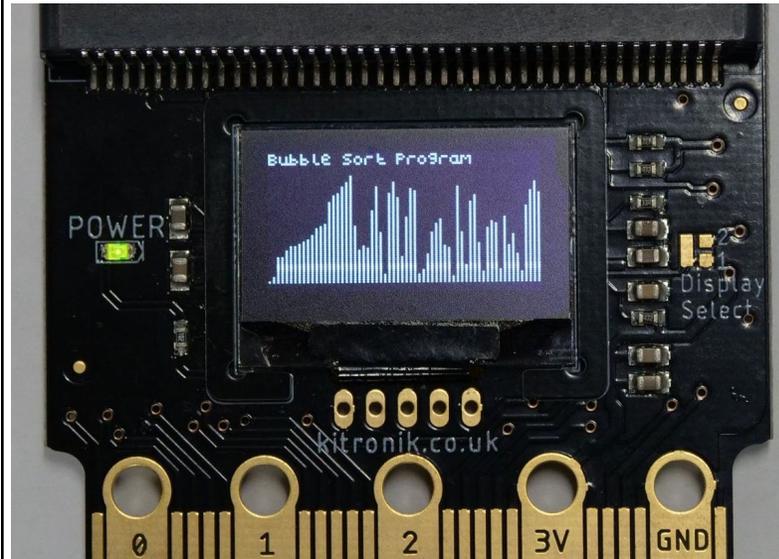
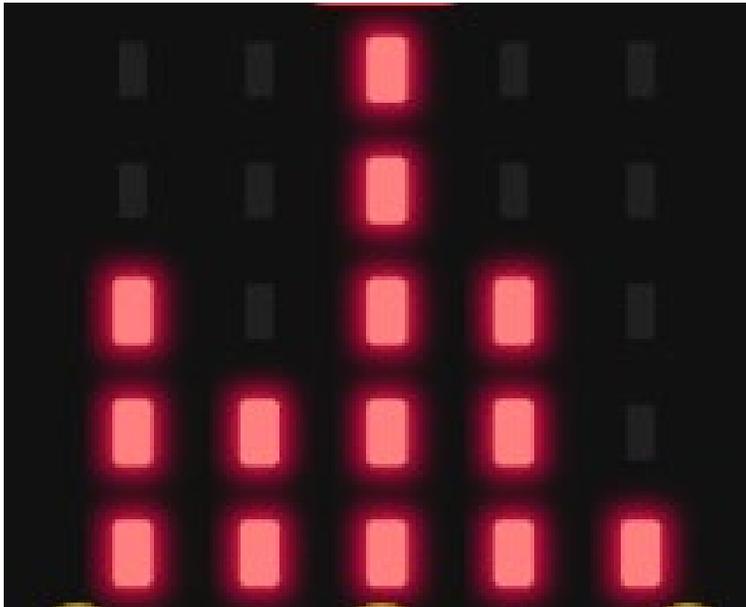
解像度

micro:bit

micro:bit + OLED

5 × 5

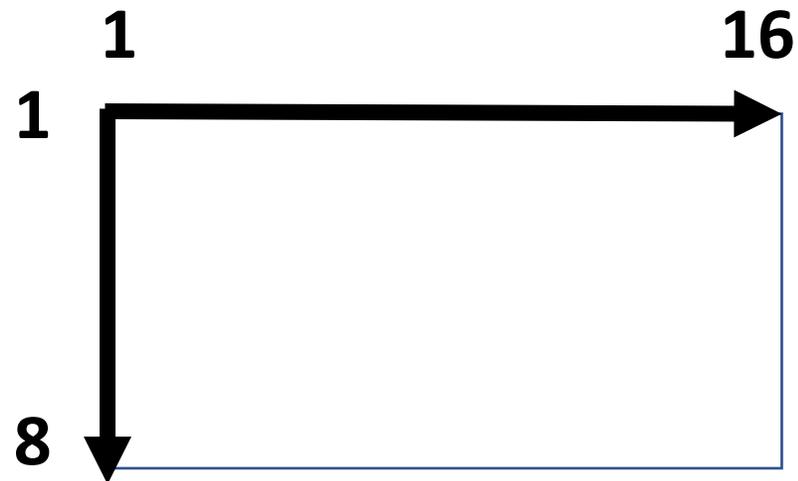
128 × 64



Kitronik :VIEW Graphics128  
OLED display 128x64

# 拡張ディスプレイでの文字の表示

16文字8行の  
英数表示が可能



最初だけ

turn  display

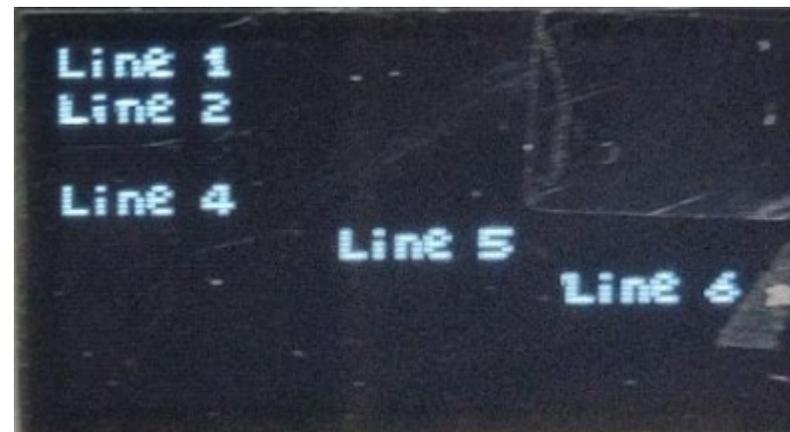
show "Line1" 

show "Line2" on line 2  

show "Line4" on line 4  

show "Line5" on line 5 with alignment: Centre  

show "Line6" on line 6 with alignment: Right  



# 拡張ディスプレイの点の座標と命令

- 命令

座標(x,y)に点を表示する

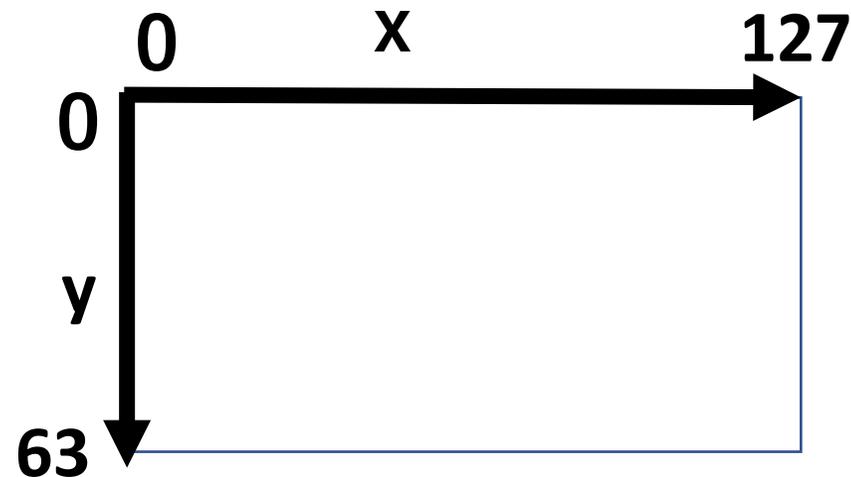
```
show pixel at x 0 y 0
```

座標(x,y)の点をクリアする

```
clear pixel at x 0 y 0
```

座標(x,y)を始点に水平/垂直な直線を描く

```
draw a horizontal line with length of 10 starting at x 0 y 0
```



128\*64dotの  
Graphicが表示可能

# 四角や線の描画のプログラム例

最初だけ

turn  display

draw a rectangle 127 wide 63 high from position x 0 y 0 ①

draw a horizontal line with length of 60 starting at x 30 y 32 ②

draw a vertical line with length of 50 starting at x 40 y 20 ③

- ① 四角形:rectangle
- ② 横線:horizontal line
- ③ 縦線:vertical line



# データの作成とその表示

ボタン A ▼ が押されたとき

変数 j を 0 ~ 63 に変えてくりかえす

リスト ▼ の j ▼ 番目の値を 0 から 50 までの乱数 にする

呼び出し グラフの表示

リスト(j)( $0 < j < 63$ )のデータは  
0 から 50 までの乱数

関数 グラフの表示 ^

clear display

show title ▼ on line 1 - +

変数 k ▼ を 0 ~ 63 に変えてくりかえす

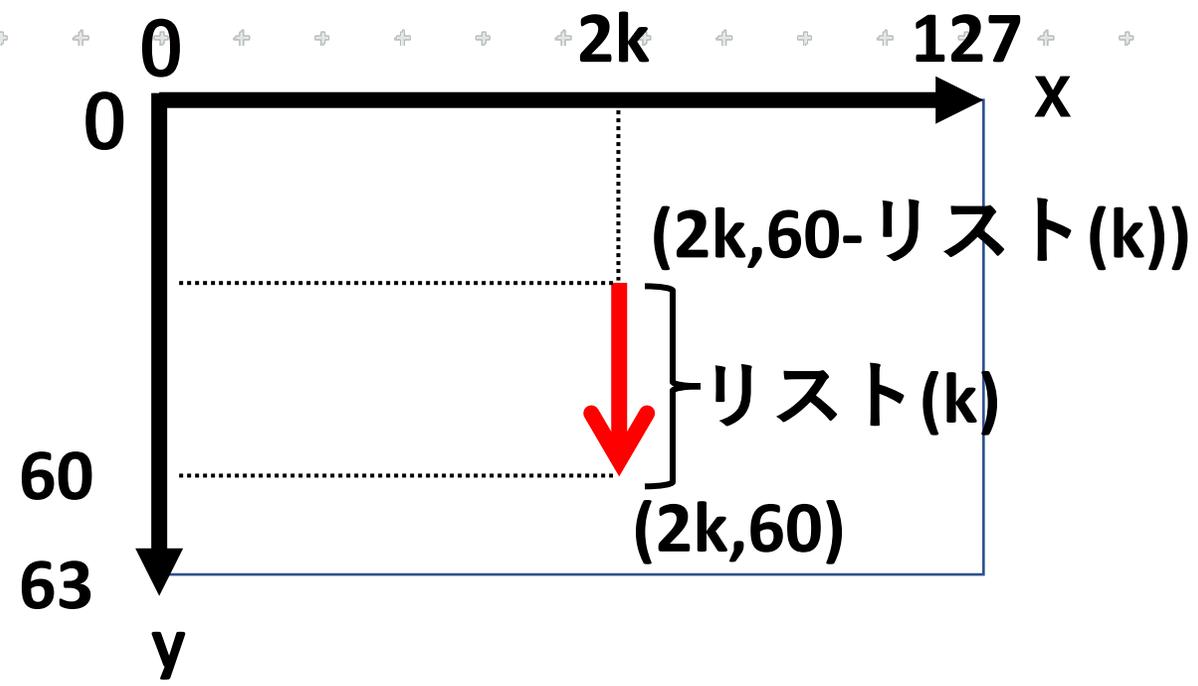
draw a vertical ▼ line with length of 0 - ▼ リスト ▼ の k ▼ 番目の値 starting at x k ▼ X ▼ 2 y 60

一時停止 (ミリ秒) 100 ▼

# 拡張ディスプレイへのデータの描画(1)

```
関数 グラフの表示 (^)  
clear display  
show title on line 1  
変数 k を 0 ~ 63 に変えてくりかえす  
draw a vertical line with length of リスト の k 番目の値 starting at x k x 2 y 60 - リスト の k 番目の値  
一時停止 (ミリ秒) 100
```

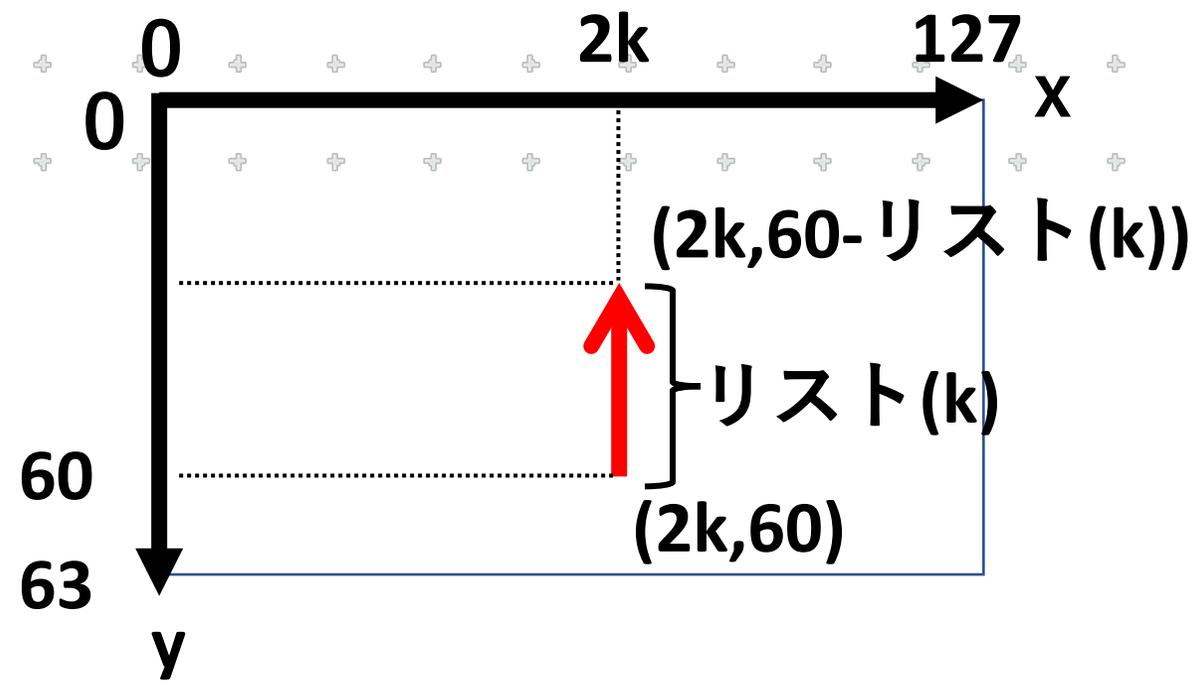
x軸方向を1つおきに設定して、y軸方向は60-リスト(k)の値からリスト(k)の長さ分の線を描く



# 拡張ディスプレイへのデータの描画(2)

```
関数 グラフの表示  
clear display  
show title on line 1  
変数 k を 0 ~ 63 に変えてくりかえす  
draw a vertical line with length of 0 リスト の k 番目の値 starting at x k x 2 y 60  
一時停止 (ミリ秒) 100
```

長さは負でも動作するので、y軸方向は負のリスト(k)の長さ分の線を描く



# Bubble Sort例

関数 Bubble

変数 並び替え範囲 を 63 にする

もし 並び替え範囲  $\geq$  1 ならくりかえし

変数 j を 0 にする

くりかえし 並び替え範囲 回

もし リストの j 番目の値 > リストの j + 1 番目の値 なら

変数 temp を リストの j 番目の値 にする

リストの j 番目の値を リストの j + 1 番目の値 にする

リストの j + 1 番目の値を temp にする

+

変数 j を 1 だけ増やす

呼び出し グラフの表示

変数 並び替え範囲 を -1 だけ増やす

A(j) がA(j+1)  
より大きいときは  
入れ替える

# Insertion Sort例

関数 Insertion

変数 並び替え範囲 を 63 にする

変数 i を 1 にする

もし 並び替え範囲  $\geq$  1 ならくりかえし

変数 temp を リスト の i 番目の値 にする

変数 j を i - 1 にする

もし j  $\geq$  0 かつ リスト の j 番目の値  $\geq$  temp ならくりかえし

リスト の j + 1 番目の値を リスト の j 番目の値 にする

変数 j を j - 1 にする

リスト の j + 1 番目の値を temp にする

変数 i を 1 だけ増やす

変数 並び替え範囲 を -1 だけ増やす

呼び出し グラフの表示

# Shaker Sort例

```
関数 shaker
  変数 並び替え範囲 を 63 にする
  もし 並び替え範囲 ≥ 1 ならくりかえし
  変数 i を 0 にする
  くりかえし 並び替え範囲 回
  もし リストの i 番目の値 > リストの i + 1 番目の値 なら
    変数 temp を リストの i 番目の値 にする
    リストの i 番目の値を リストの i + 1 番目の値 にする
    リストの i + 1 番目の値を temp にする
  +
  変数 i を 1 だけ増やす
  変数 j を 63 にする
  くりかえし 並び替え範囲 回
  もし リストの j 番目の値 < リストの j - 1 番目の値 なら
    変数 temp2 を リストの j 番目の値 にする
    リストの j 番目の値を リストの j - 1 番目の値 にする
    リストの j - 1 番目の値を temp2 にする
  +
  変数 j を -1 だけ増やす
  呼び出し グラフの表示
  変数 並び替え範囲 を -1 だけ増やす
```

先頭→最後尾 でデータ交換を行う

最後尾→先頭 でデータ交換を行う

# 内部タイマーを使ったソートの稼働時間の表示



図3 内部タイマーを使ったソートの稼働時間の計算方法とその表示プログラム

micro:bitには内部に時計機能があり、それを使って時間を計測できる。ただし、稼働時間は電源ONにしてからの時間

稼働時間 (ミリ秒)

稼働時間 (マイクロ秒)

# ソート動画のデモンストレーション

# 結果および考察

高等学校情報 I のアルゴリズムとプログラミングの単元で, micro:bit に拡張ディスプレイを接続して, **整列のプログラム** を作成した.



拡張ディスプレイがあることで, **変数の値の可視化が可能** になり, **プログラム全体の変化が見えて, 動作を大局的に俯瞰** できる.