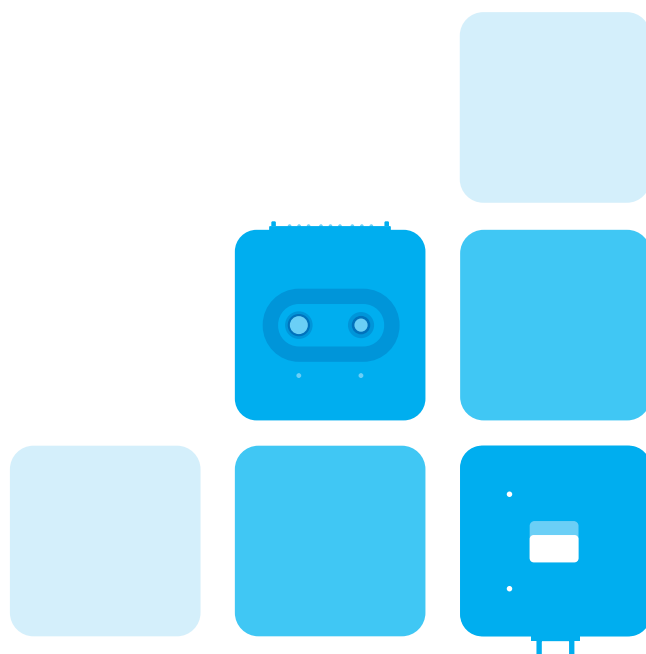
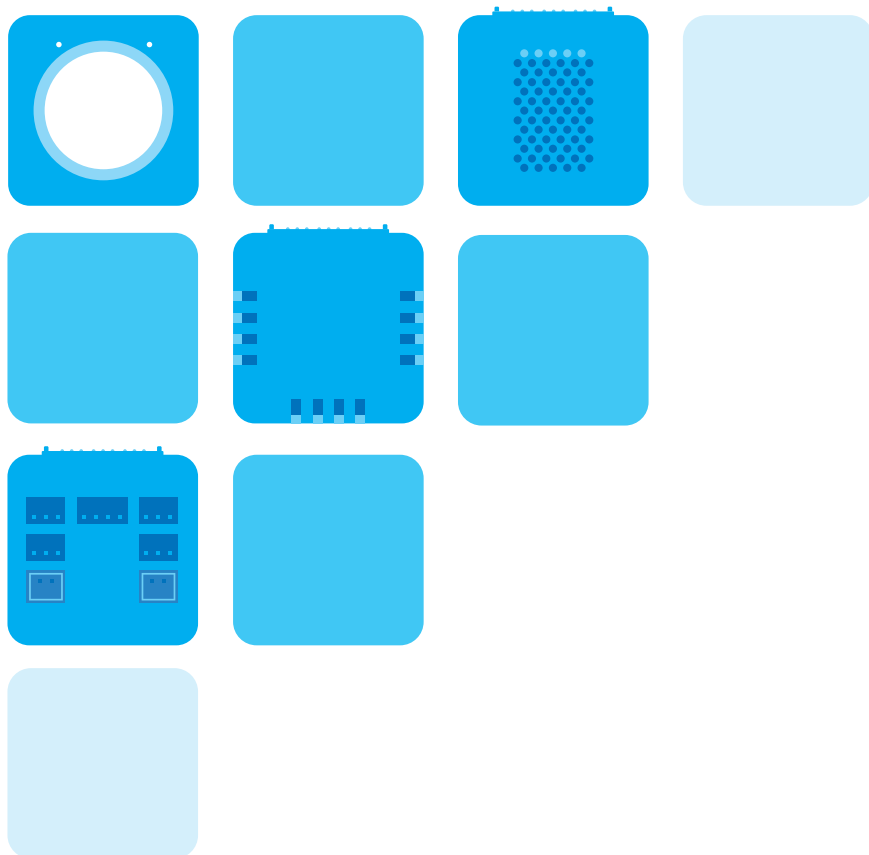


ArteC[®] Links

アーテックリンクス

基本セット

教員用



学習内容

- 「プログラム」と「プログラミング」について
- フローチャートの書き方

達成目標

- 「プログラム」と「プログラミング」の違いを説明できるようになる
- プログラミングの際に意識すべきことを理解する
- フローチャートを書けるようになる
- フローチャートを書くことの利点を理解する

活動内容	評価
導入	
●「プログラム」と「プログラミング」…………… P.2	知識及び技能①
●手順やルールをコンピューターが理解するためには…………… P.2	知識及び技能①
基本課題① 「歩く」をプログラミングしよう	
●「歩く」という動作をプログラミングしてみましょう…………… P.3	知識及び技能②
基本課題② 「歩く」をフローチャートにあらわそう	
●フローチャート…………… P.4	知識及び技能②
●「4歩歩く」をフローチャートにあらわす…………… P.6	知識及び技能②
まとめ	
●「プログラム」と「プログラミング」…………… P.7	知識及び技能①
●手順やルールをコンピューターが理解するためには…………… P.7～8	思考・判断・表現

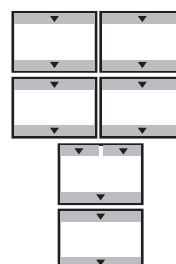


アイコンシート ご活用ください！

生徒がカードを並べて課題解決のヒントにできる補助教材です。

プログラムの

- ①アイデアを整理できる
- ②流れを考えられる
- ③段階的に考えられる



無料
ダウン
ロード



アイコンシート：ダウンロード URL : <https://www.artec-kk.co.jp/arteclinks/school/juniorhigh/>

基本

プログラムとプログラミング

年 組 番 名前

導入

知識技能①

i 「プログラム」と「プログラミング」

「プログラム」について

「プログラム」とは、コンピューターに指示を出すための手順やルールをまとめたものです。これを使ってコンピューターを制御し、様々な動作や処理を実行させることができます。例えば、ゲームをつくる、ウェブサイトを構築する、ロボットを制御するなど、様々なことができます。

「プログラミング」について

「プログラミング」とは、「プログラム」をつくることです。
「プログラミング」では、プログラミング言語というコンピューターが理解できる特殊な言語を使います。

```
f ifNil: [
    "Turn setting language"
    self setLanguage: ltmp.
    + self].

" Head and Footer file "
" Default English. "
headCode + StandardFileStream new open: 'code/head+en.txt' forWrite: false.
footCode + StandardFileStream new open: 'code/foot+en.txt' forWrite: false.

((ltmp = 'ja') | (ltmp = 'ja+HIRA')) ifTrue: [
    Transcript show: '----- Trans to Japan -----'; cr.
    headCode + StandardFileStream new open: 'code/head.txt' forWrite: false.
    footCode + StandardFileStream new open: 'code/foot.txt' forWrite: false.
```

〈プログラムの例〉

知識技能①

i 手順やルールをコンピューターが理解するためには

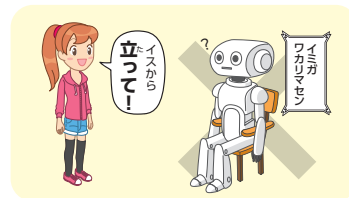
コンピューターを動かすためには、プログラムによってコンピューターに指示をしなければなりません。しかし、人間同士のように指示をしてもコンピューターは理解することができません。コンピューターに指示をする場合には、指示を「細かく・具体的」にするようにしましょう。

例) イスから立ち上がる

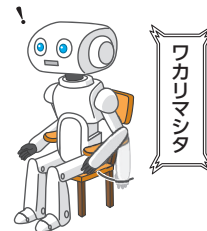
〈人間同士の場合〉



〈ロボットの場合〉



1. 両手をひじ掛けに置く
- ↓
2. 肘を伸ばして、体を座面から持ち上げる
- ↓
3. 重心を前に移動する
- ↓
4. 両手をひじ掛けから離す
- ↓
5. 両膝をまっすぐにのぼす



基本課題① 「歩く」をプログラミングしよう

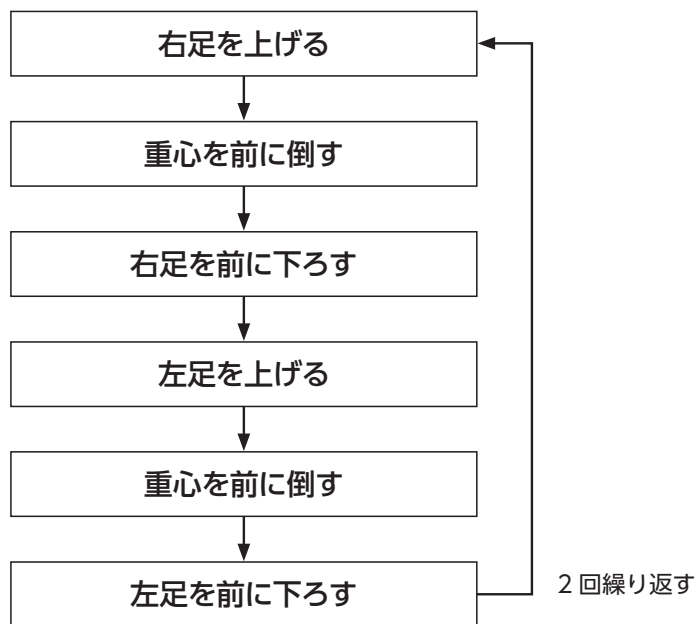
知識技能②

「歩く」という動作をプログラミングしてみましょう

ロボットに「4歩歩く」という動作を指示するためのプログラムを考えましょう。

下の空欄に書いてください。□と→は必要な分だけ書き足してください。

(解答例)



【狙い】

- プログラミングでは、指示が「詳細に・具体的に」書かれている事が大切である事に気づく。
- 自分の身体を使って、プログラムを体験することで、プログラミングを身近に感じてもらう。

【課題】

ロボットに「4歩歩く」という動作を指示するためのプログラムを考えます。

「プログラムをコンピューターやロボットが理解するために必要なこと」で示した「イスから立ち上がる」ためのプログラムを参考にして、「4歩歩く」ためのプログラムを考えます。

プログラム完成後には、実際にそのプログラムに沿って歩いてみるよう促します。実際に歩いてみて、改善点を探し、プログラムを改善するという「デバッグ」の流れも体感できるようにします。

デバッグとはプログラム上の誤り（バグ）を発見して訂正すること。

知識技能①

基本課題② 「歩く」をフローチャートにあらわそう



フローチャート

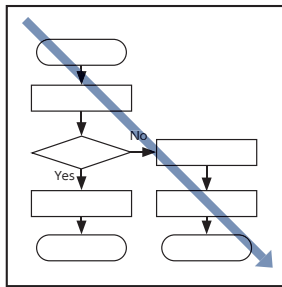
「フローチャート」とは、記号を使って、処理や動作の手順をわかりやすく示すための方法です。

「フローチャート」を書くことによって、手順やルールをわかりやすく整理することが出来、プログラミングの手助けをします。

また、フローチャートには図と記号の使い方のルールが定められています。そのため、だれが見ても理解することができ、自分の考えたプログラムを相手に伝えることに適しています。

フローチャートのルール ①

処理や動作の流れは、「上から下へ」かつ「左から右へ」流れるように要素を配置します。

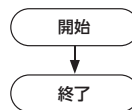


フローチャートのルール ②

各記号には決まった意味と用途があり、適したものを使用します。

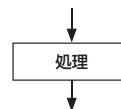
端子

フローチャートのスタートとゴールを意味する記号です。

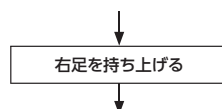


処理

動作を記入する記号です。四角のなかに動作の内容を記入しましょう。
一つの四角につき、一つの動作を記入します。



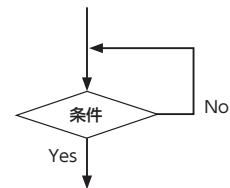
例) 右足を持ち上げる動作



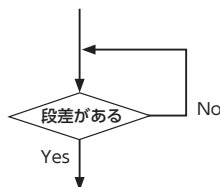
知識技能①

条件分岐

条件を記入する記号です。ひし形の中に分岐を行うための条件を記入しましょう。
その条件に適する場合は「Yes」か「真」と書かれた矢印を伸ばします。適さない場合は「No」か「偽」と書かれた矢印を伸ばします。



例 歩く先に段差があるかどうかで動作を変えたい



繰り返し

繰り返し処理を記入する記号です。台形のような六角形の図形の間にある処理が繰り返されます。繰り返す回数を記号内に書くようにしましょう。



例 4 回足踏みをする動作

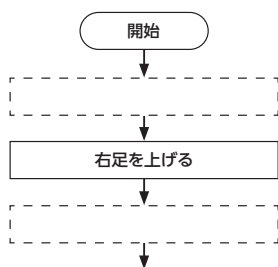


知識技能②

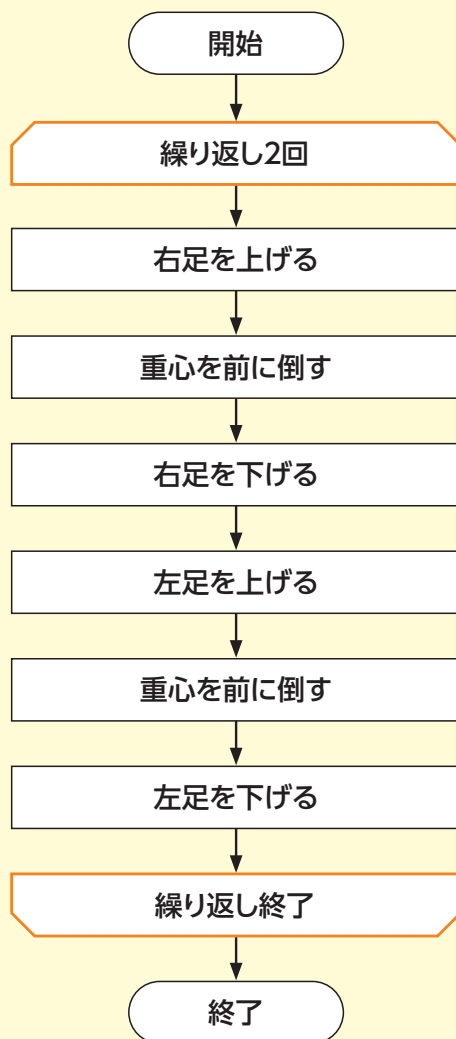
「4歩歩く」をフローチャートに表しましょう

P. 2で作成したプログラムをフローチャートで表してみましょう。

「フローチャートのルール」に従って、下の空欄に書いてみましょう。記号と矢印は必要な分だけ書き足してください。



解答例



まとめ

知識技能①

「プログラム」と「プログラミング」

下の赤枠に適した語句を入れましょう。

- ①コンピューターに指示を出すための手順やルールをまとめたものをなんと言うか。



プログラム

- ②コンピューターへの指示をつくることをなんと言うか。



プログラミング

- ③プログラミングで使う特殊な言語のことをなんと言うか。



プログラミング言語

5

思考・判断・表現

手順やルールをコンピューターが理解するためには

下の赤枠に「4 歩歩く」のプログラムを作成した際に意識したこと、注意したことについて書きましょう。

解答例

プログラムの作成をする際には、指示の内容を具体的にするように気を付けました。
足の動きだけではなく、膝の動きといった部分にも注意しました。
また、自分の書いたプログラムを実行してみると、もっと細かい指示が必要である
ことがわかったので、自分が作成したプログラムを修正しました。

〈記述内容別観点〉

・プログラムを作成する際には、指示の内容を具体的に・詳細にすることが大切だという内容が含まれている。

→思考力・判断力・表現力

・プログラムを実行して、改善点を発見したという内容が含まれている。

→学びに向かう力・人間性等

知識技能②

フローチャート

下の赤枠に適した語句や説明を入れましょう。

①右の記号はフローチャートのどの部分で使われますか？

→ フローチャートのスタートとゴールで使用する。

②右の記号はどんな時に使われますか？

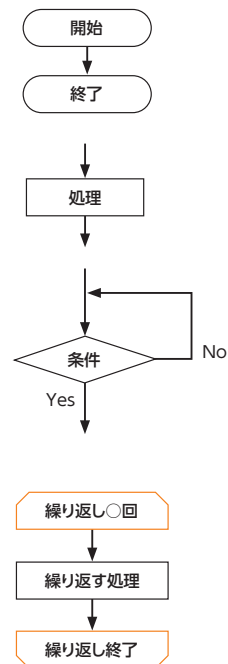
→ プログラムの動作を作る際に使用する。

③右の記号はどんな時に使われますか？

→ 条件分岐をつくる際に使用する。

④右の記号はどんな時に使われますか？

→ 繰り返し処理をつくる際に使用する。



学習内容

- ブロックプログラミングの使い方
- メインユニットの接続方法
- 順次処理・繰り返し処理・条件分岐処理

達成目標

- 身のまわりにはどのようなプログラムで制御されている製品があるのかに気づく。
- 信号機を再現するプログラムを作成する。
- 順次処理・繰り返し処理・条件分岐処理について理解する。
- プログラムの組み方を理解する。

活動内容	評価
導入	
●社会で使われている信号機とその工夫…………… P.10	知識及び技能①
●身のまわりの製品で使われているプログラムを考えよう…………… P.12	知識及び技能①
基本課題① メインユニットの明かりをつけよう	
●接続方法…………… P.13	知識及び技能②
●メインユニットの明かりをつけよう…………… P.14	知識及び技能②
●LED ライトの色を変えよう…………… P.14	知識及び技能②
基本課題② 自動車用信号機を作成しよう	
●順次処理…………… P.16	知識及び技能①
●自動車用信号機を作成しよう…………… P.16	知識及び技能②
基本課題③ 歩行者用信号機を作成しよう	
●繰り返し処理…………… P.19	知識及び技能①
●歩行者用信号機を作成しよう…………… P.19	知識及び技能②
基本課題④ 押しボタン式信号機を作成しよう	
●条件分岐処理…………… P.22	知識及び技能①
●押しボタン式信号機を作成しよう…………… P.22	知識及び技能②
まとめ	
●用語の理解…………… P.23	知識及び技能①

基本

身のまわりの計測と制御

年 組 番 名前

知識技能①



社会で使われている信号機とその工夫

社会では様々な信号機が使用されています。信号機の重要な役割として次のことが挙げられます。

- ① 交通事故の防止
- ② 車の流れをスムーズにする
- ③ 交通環境の改善

信号機はこのような役割を担っていて、この役割を達成するために様々な工夫がされています。

例えば車が来ているかどうかで信号の色を変える「感应式信号機」は、入力装置・制御装置・出力装置の3つの要素から構成されています。

●入力装置（センサー）…………… センサーは、周囲の状況や情報を感知し、それをデータに変換する装置です。
例えば、気温、明るさ、音の大きさ、物体の接触など、様々な情報を収集できます。センサーは自動車、スマートフォン、気象観測などさまざまな場面で利用されています。

●制御装置（コンピューター）…………… コンピューターは、情報を処理し、計算する電子装置です。コンピューターはセンサーからデータを受け取ったり、アクチュエーターに命令を出したりする能力を持っています。

●出力装置（モーター、LED など）… アクチュエーターは、命令を受けて、物理的な動作や操作を実行する装置です。例えば、物体を回転させたり移動させたりするモーターや、LED も出力装置の一つです。

例 感应式信号機



このように、感应式信号機はセンサーが周りの交通状況を読み取り、コンピューターが判断して、アクチュエーター（LED）を制御しています。この一連の動きは、コンピューターに保存された**プログラム**によって動かされています。この他の信号機も**プログラム**によって制御されています。

コラム 交通整理の三種の神器

身のまわりにある信号機には、安全に快適に通行できるように様々な工夫が施されています。そのうち、代表的な3つの工夫を紹介します。

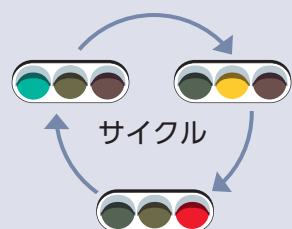
① サイクル

信号灯が青・黄・赤と一巡する時間をサイクルと言います。

サイクルが短い → 通行できる量が少なく、渋滞の原因になる

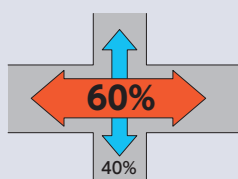
サイクルが長い → 誰も通らない無駄な時間が増える

サイクルの長さは、交通量・交差点の大きさ、歩行者の横断時間などを考慮して決定されています。



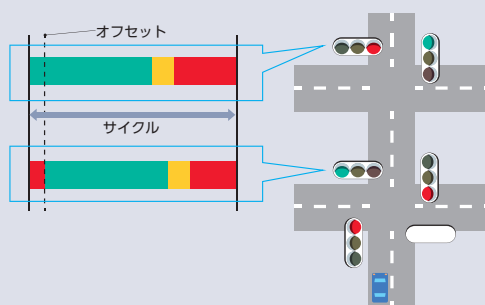
② スプリット

1回のサイクルのうち、青・黄・赤の灯りの時間配分のことを言います。交通量の多い幹線道路とその道路と交差する小さな道路がある場合、青信号の長さが同じだと無駄が生じてしまいます。そのため、各信号の点灯時間を調整して、交通量の多い道路は青信号を長くするなどの調整を行います。



③ オフセット

幹線道路で、信号に止められずにスムーズに走行できるように青信号になるタイミングをずらすことを言います。

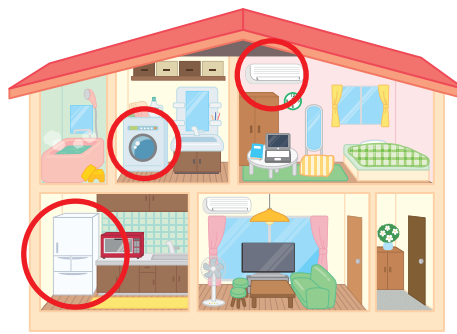
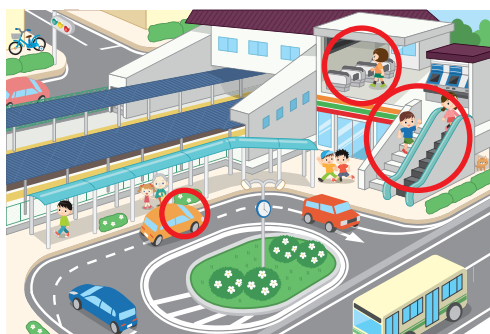




身のまわりの製品で使われているプログラムを考えよう

下の二つの絵から、プログラムによって制御されている製品を探してみましょう。

また、その製品にはどのような工夫がされているでしょうか。下の表に書き出してみましょう。



思考・判断・表現

製品名	工夫
例 冷蔵庫	例 冷蔵庫内の温度をセンサーで測定し、庫内を適した温度に保つようにする。
自動改札機	きっぷ（や IC カード）の磁気をセンサーで検知し、改札の扉を開けるようにする。
エアコン	部屋の温度をセンサーで測定し、設定温度に近づけるようにエアコンを制御する。
エスカレーター	人が通ったことをセンサーで検知し、エスカレーターを動かす。
カーナビゲーション	今いる位置を測定し、目的地までの経路を考える。
洗濯機	洗濯物の重量を測定して、最適な洗剤の量を投入する。

基本

身のまわりの計測と制御

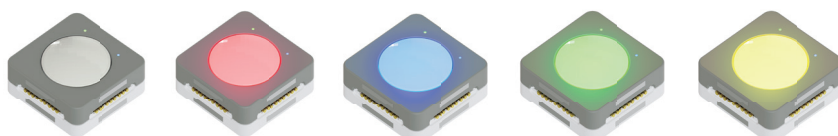
年 組 番 名前

知識技能②

基本課題① メインユニットの明かりをつけよう

「メインユニット」は、プログラムによって動く小さなコンピューターです。また、プログラムによって LED を点灯させるなどの命令を実行することができます。

白い部分は押しボタンになっています。また、フルカラーの LED も中に組み込まれています。

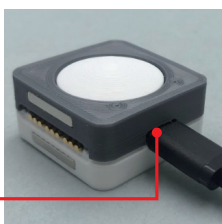


接続方法

1. ソフトウェアを起動させます
2. メインユニットとパソコンを接続します。

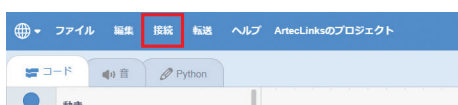


メインユニット側

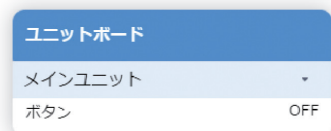


パソコン側

3. 画面左上のタブから「接続」を選択し、「USB JTAG…」で始まる URL をクリックし、「接続」を選択します。



4. 「ユニットボード」が画面上に表示されたら接続完了です。

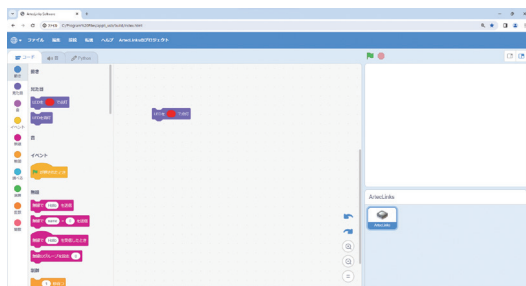


知識技能②

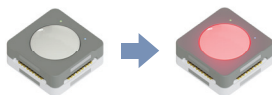
メインユニットに明かりをつけよう


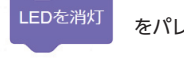
接続完了後、メインユニットを光らせてみましょう。

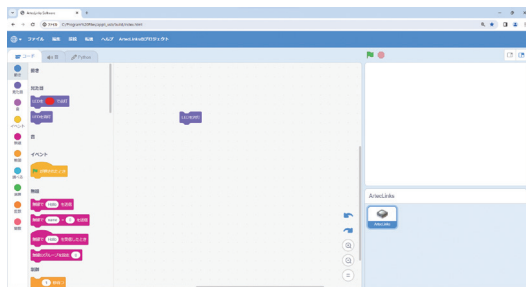
- ①カテゴリ内の  から  をパレット上にドラッグ&ドロップし、クリックします。



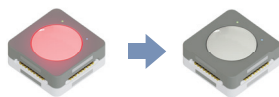
- ②メインユニットのLED が赤く点灯していることを確認します。





- ③カテゴリ内の  から  をパレット上にドラッグ&ドロップし、クリックします。



- ④メインユニットのLED が消灯していることを確認します。



LEDライトの色を変えよう

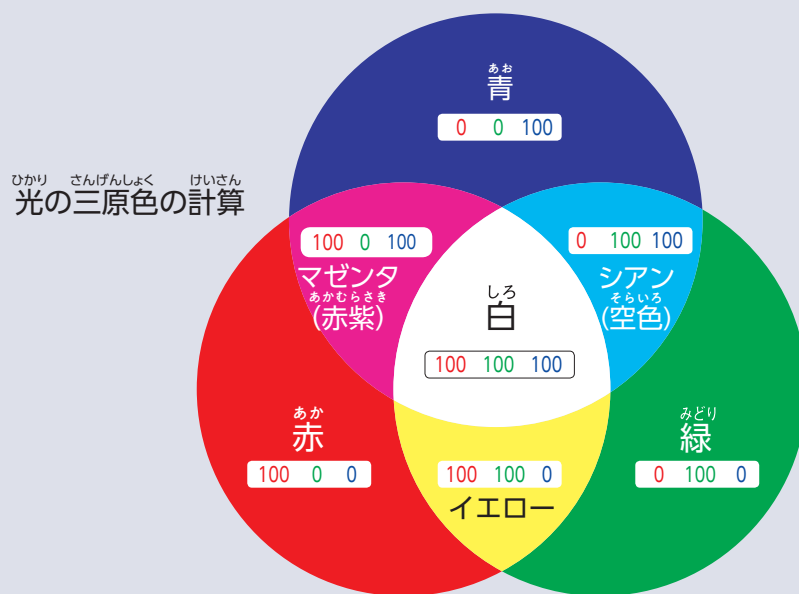
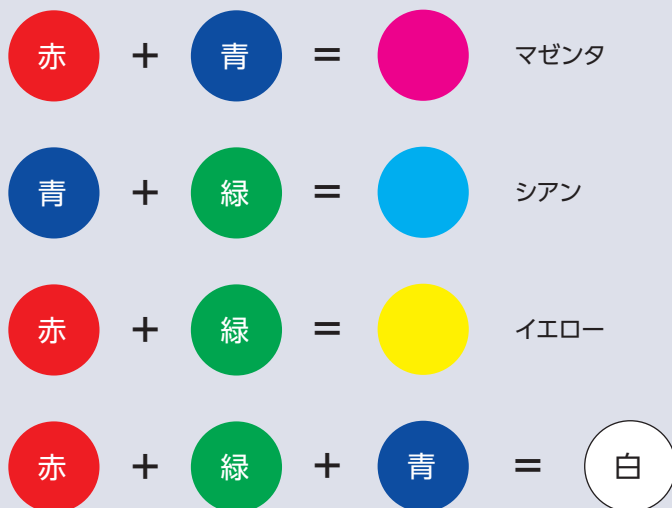
ブロックの  の部分をクリックすると、右のような表示になります。それぞれの色の  をスライドさせると、色を変えることができます。



コラム 加色混合

加色混合は、色を組み合わせる新しい色を作る方法のことを指します。これは、光を使って色を混ぜる方法です。赤・緑・青の三つの色を用いて、様々な色を作ることができます。

【色の組み合わせ方】



メインユニットの LED も同様に赤・青・緑の光の強さを 0～100 で制御しています。

基本

身のまわりの計測と制御

年 組 番 名前

基本課題② 自動車用信号機を作成しよう

知識技能①

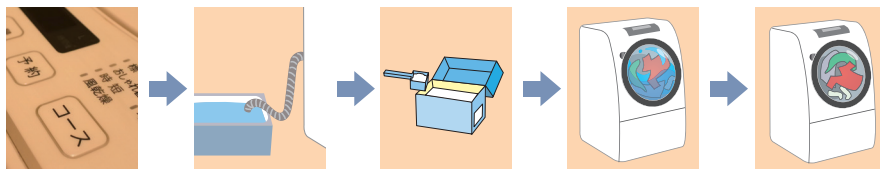


順次処理

「順次処理」とは、命令や作業を一つずつ順番に行っていく処理方法です。

順次処理はプログラムの基本的な処理方法です。下の洗濯機のように、一つの作業が完了してから、次の作業に向かうような処理のことを「順次処理」と言います。

例 洗濯機



洗濯コースの決定

水を入れる

洗剤を入れる

洗濯機をまわす

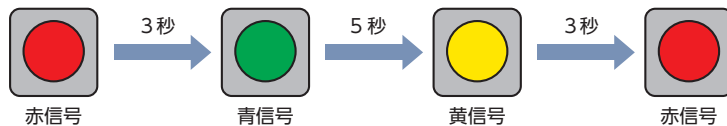
乾燥させる

知識技能②

自動車用信号機を作成しよう

次の自動車用信号機の動作をプログラミングしましょう。

【動作】



赤信号

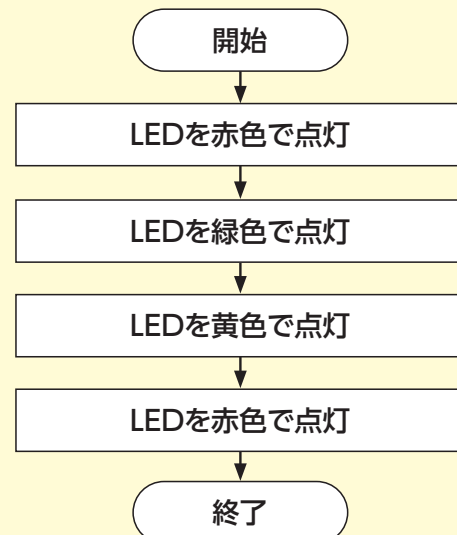
青信号

黄信号

赤信号

①フローチャートを考え、下の赤枠内に書き入れましょう。

解答例





ブロックプログラミング

ブロックプログラミングの基本的なやり方について紹介します。

ブロックプログラミングとは、ブロックをつなげてプログラミングしていくことを指します。ブロック 1 つで 1 つの命令が実行されます。

以下では、ブロックプログラミングの基本的なルールを紹介します。

ルール 1 プログラムの一番初めには必ずハットブロックをつける

ハットブロックとは、以下のようにブロックの上に楕円がついているブロックのことを指します。



このブロックをプログラムの一番上につけるようにします。ハットブロックはこの下につなげられたプログラムを実行する機能を持ちます。ハットブロックをつけていないとプログラムが実行されません。



ルール 2 ブロックをつなげるように配置する

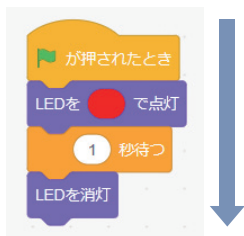
プログラムを作成するときには、ブロックをつなげるようにしてください。

以下のように離れていると、プログラムを実行することができません。



ルール 3 プログラムは上から下に実行される

作成されたプログラムは、上から下に向かって実行されます。



② ①で作成したフローチャートを参考にして、プログラムを作成しましょう。以下の「使用ブロック」………を参考にしてください。

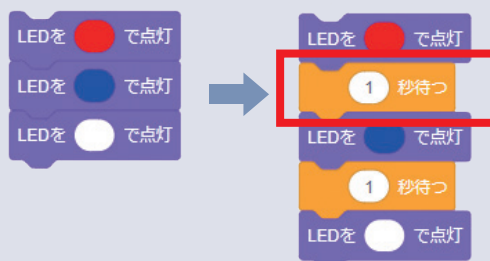
解答例



こんな時どうする？

● LEDが一瞬で光ってしまい、白の光しか光らない

→プログラムは一瞬で行われてしまうので、LEDを光らせるブロックの間に「x秒待つ」というブロックを入れる必要があります。



こんな時どうする？

● プログラムが終了したのに、LEDが消えない

→プログラムが終了したときには、プログラムの最後の動作が継続されます。そのため右のプログラムでは、LEDを消灯するブロックが継続されてしまいます。

LEDを消灯するためには、LEDを消灯をクリックするか、右プログラムのよう「LEDを消灯」ブロックをプログラムに組み込みます。



知識技能①

基本

身のまわりの計測と制御

年 組 番 名前

基本課題② 歩行者用信号機を作成しよう

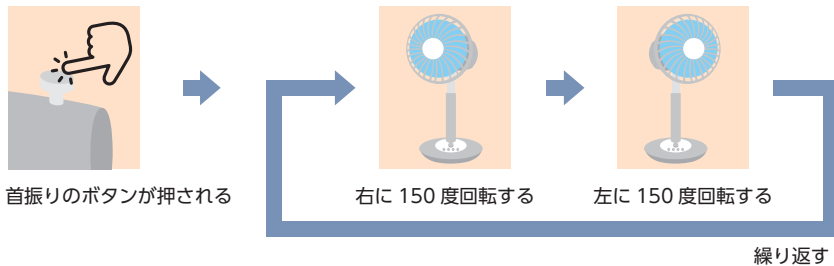


繰り返し処理

「繰り返し処理」とは、同じ内容の作業や命令を何度も繰り返して行うための処理方法です。

繰り返し処理を用いることによって、同じプログラムを何度も作る必要がなく、効率化や作業の自動化を行うことができます。下の扇風機のように繰り返し処理を行うことで、2つの命令のみで扇風機の首振りをプログラミングすることができます。

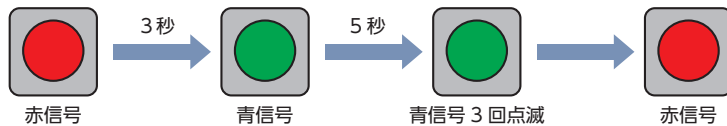
例 扇風機



歩行者用信号機を作成しよう

次のような歩行者用信号機をプログラミングしましょう。

【動作】



知識技能②

①フローチャートを考え、下の赤枠内に書き入れましょう。

15

解答例





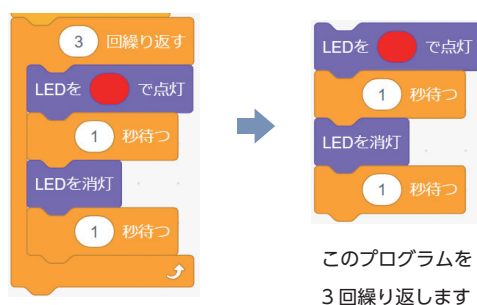
繰り返しブロック

繰り返し処理を行うためには、繰り返しブロックを使う必要があります。

繰り返しブロックの間に挟まれているブロックのプログラムを指定した回数だけ繰り返し、実行します。

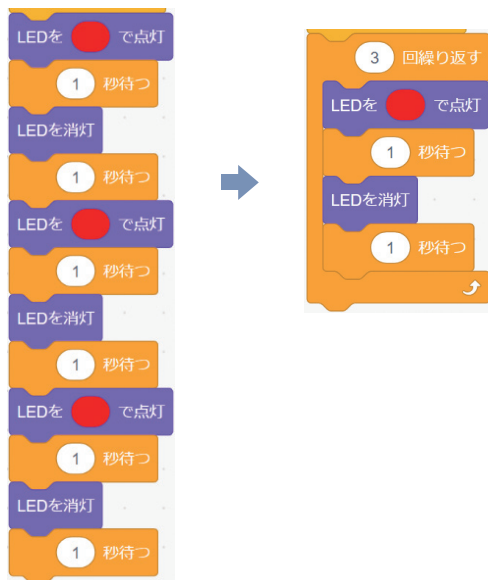


例 赤色 LED を 3 回点滅させるプログラム



繰り返し処理を用いない場合、以下のようにプログラムが長くなってしまいます。

プログラムはできるだけ簡潔に、短く組めるよう工夫することが大切です。そうすることでデバッグ（プログラムのバグを修正する作業）がやりやすくなったり、バグの発生を防いだりすることができます。



知識技能②

② ①で作成したフローチャートを参考にして、プログラムを作成しましょう。以下の「使用ブロック」を参考に
してください。

プログラ
ミング
実習

解答例：点滅



知識技能②

② ①で作成したフローチャートを参考にして、プログラムを作成しましょう。以下の「使用
ブロック」を参考にしてください。

プログラ
ミング
実習

解答例：歩行者用信号機



基本

身のまわりの計測と制御

年 組 番 名前

基本課題③ 押しボタン式信号機を作成しよう

知識技能①



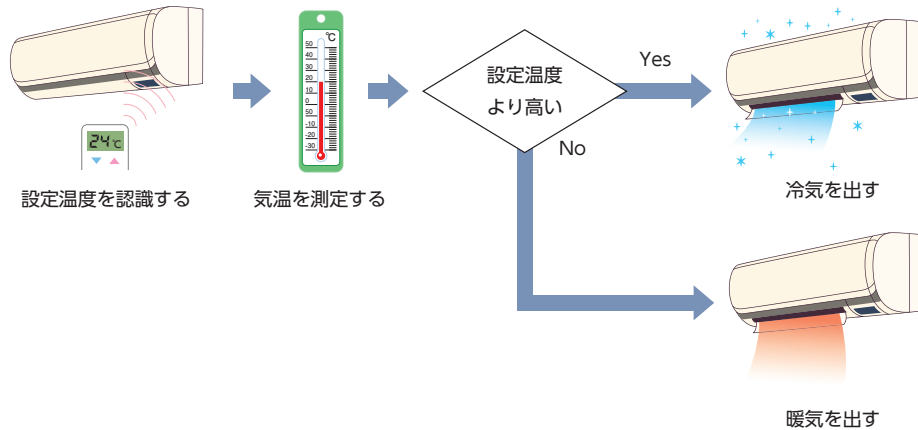
条件分岐処理

「条件分岐処理」とは、設定された条件が満たされているかどうかを確認し、条件が満たされている場合と満たされていない場合で、異なる作業や命令を実行するための処理方法です。

条件分岐処理を用いることで、その時その時の状況に合わせたプログラムを実行することが可能となります。

下のエアコンのように、条件分岐処理を用いることでその時の状況にあった命令や作業を選択することができます。

例 エアコン

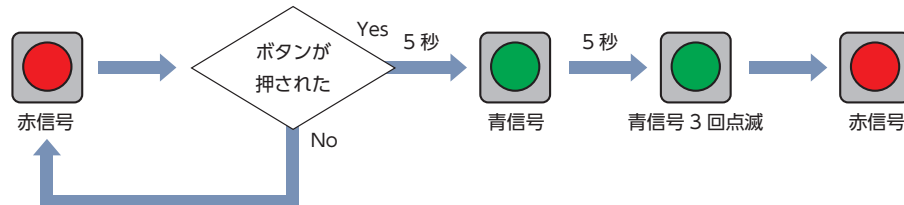


知識技能②

押しボタン式信号機を作成しよう

次のような押しボタン式信号機をプログラミングしましょう。

【動作】



知識技能②

①フローチャートを考え、下の赤枠内に書き入れましょう。

知識技能①



条件分岐ブロック

条件分岐処理を行うために、条件分岐ブロックを使います。

●条件分岐ブロック

条件が満たされている場合、条件分岐ブロックの間に挟まれているブロックのプログラムを実行します。



●条件ブロック

条件分岐ブロックで用いる条件を設定するためのブロックです。以下のような形をしたブロックが当てはまります。



「ある値が 50 を超えているかどうか」という条件を設定する



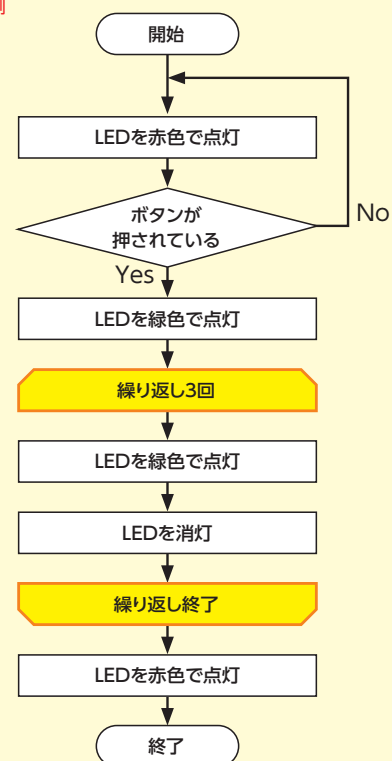
「ボタンが押されているかどうか」という条件を設定する

例 メインユニットのボタンが押されたときに、1 秒間 LED ライトを点灯するプログラム



「メインユニットのボタンが押された」
ときに実行されるプログラム

解答例



基本

身のまわりの計測と制御

年 組 番 名前

知識技能①



繰り返しブロック(ずっと～する)

プログラムを何回も実行させるためには「ずっと～する」という繰り返しブロックを使います。

●繰り返しブロック

このブロックの間に挟まれているブロックのプログラムを何回も実行します。



例 メインユニットのボタンが押されたときに、LED ライトを 1 秒間点灯するプログラム



「ボタンが押されているか」を判定するためのプログラム]
このプログラムを何回も実行させる必要があります。

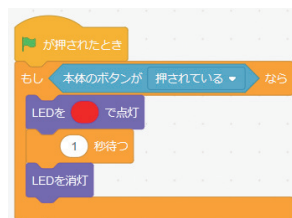
プログラムの実行は一瞬で行われます。そのため、条件が満たされているかどうかの判定を常に行うためにはプログラムを何回も実行させなければなりません。

そこで、「ずっと～する」の繰り返しのブロックを使います。

- 「ずっと～する」のブロックがあるため、ボタンが押されているかどうかの判定を何度も実行することができます。



- ✗ プログラムの実行と同時にプログラムが終了してしまいます。そのため、ボタンが押されているかどうかの判定を常に行うことができません。



知識技能①

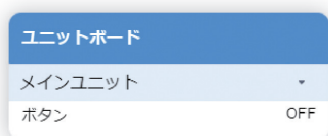


センサーの値を読み取る方法

条件分岐の条件には、センサーの値を使うことができます。

例えば、押しボタン式信号機では、「ボタンが押されているか」を判定するために、ボタンの状態を条件として使っています。

メインユニットのボタンが押されているときと押されていない時の表示の変化をユニットボードで確認し、赤枠内に書き入れましょう。



ボタンが押されているとき



ON

ボタンが押されていないとき



OFF

プログラミング
実習

② ①で作成したフローチャートを参考にして、プログラムを作成しましょう。以下の「使用ブロック」を参考にしてください。

知識技能②

使用ブロック



メインユニットの LED を光らせるためのブロックです。



メインユニットの LED を消灯するためのブロックです。



次の処理の実行を遅らせるためのブロックです。指定した秒数



ずっと繰り返して実行するためのブロックです。このブロックのプログラムをずっと繰り返し実行します。



条件分岐を行うためのブロックです。条件が満たされたまれているブロックのプログラムを実行します。



繰り返し処理を行うためのブロックです。このブロックのプログラムを指定回数繰り返します。



条件を設定するためのブロックです。条件の部分に入れることができます。「押されている」以外の条件を設定することもできます。

解答例



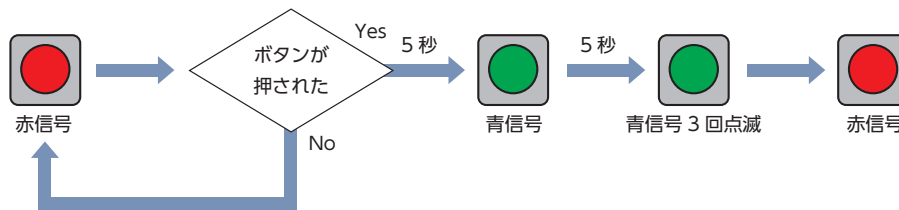
基本

身のまわりの計測と制御

年 組 番 名前

③プログラムが完成したら、ソフトウェア画面右上の  を押して、プログラムを実行しましょう。

④プログラムが正しく動作しているのか確認しましょう。



⑤達成出来た項目にチェックを入れましょう。

- ☐ 「条件分岐処理」がどのような処理方法であるか理解できた。
- ☐ フローチャートを書くことができた。
- ☐ センサーの値を読み取ることができた。
- ☐ 押しボタン式信号機をプログラミングすることができた。

まとめ

知識技能①

以下の文に当てはまる適切な語句を、それぞれの赤枠内に書き入れましょう。

①順番通りにプログラムが実行される処理方法の名前

➡ 順次処理

②同じプログラムを繰り返す処理方法の名前

➡ 繰り返し処理

③条件にあわせて、実行するプログラムが変更される処理方法の名前

➡ 条件分岐処理

学習内容

- 変数、リスト、無線通信
- 情報通信ネットワーク
- アクティビティ図

達成目標

- 変数、リスト、無線通信の仕組みの理解
- 情報通信ネットワークとはどのようなものなのかの理解
- アクティビティ図の作成方法の理解

活動内容	評価
基本課題① 反応時間計測タイマーを作成しよう	
●変数…………… P.28	知識及び技能①
●反応時間計測タイマーを作成する…………… P.29	知識及び技能②
●タイマーブロック…………… P.32	知識及び技能①
●乱数ブロック…………… P.33	知識及び技能①
基本課題② 早押しボタンを作成しよう	
●情報通信ネットワーク…………… P.35	知識及び技能①
●リスト…………… P.36	知識及び技能①
●アクティビティ図…………… P.38	知識及び技能①
●無線ブロック…………… P.43	知識及び技能①
●リストブロック…………… P.45	知識及び技能①
まとめ	
●まとめ…………… P.50	知識及び技能①

基本

早押しボタンを作成しよう

年 組 番 名前

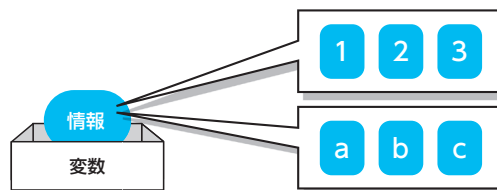
基本課題① 反応時間計測タイマーを作成しよう

知識技能①

i 変数

「変数」とは数字や文字などの情報を保存するための箱のようなものです。プログラム上で情報を保存するために使われます。

変数には、数字や文字を「ひとつだけ」保存させることができます。



変数は一つの情報しか保存できないため、新しい情報を保存させると古い情報は上書きされます。



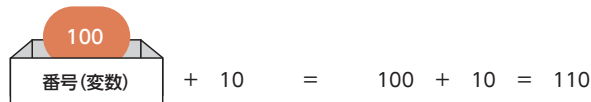
中に入っている情報が何かわかりやすくするために、変数には名前を付けることができます。

その名前を呼び出すことで、その変数の中に入っている情報をプログラムの中で活用することができます。

例 ①ある変数に「番号」という名前をつけ、そこに「100」を保存する。



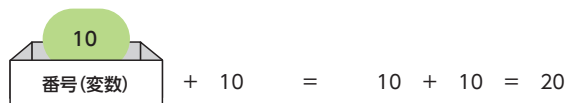
②『「番号」 + 10』というプログラムを作成した場合、「110」という答えが出る。



③変数「番号」の中身を、「100」から「10」に入れ替える。



④『番号』+ 10』というプログラムを作成した場合、『20』という答えが出る。

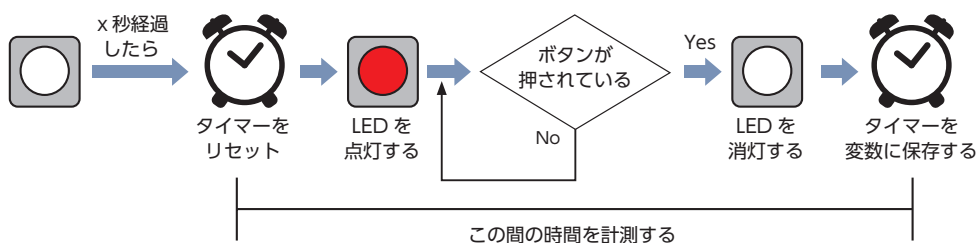


知識技能②

反応時間計測タイマーを作成しよう

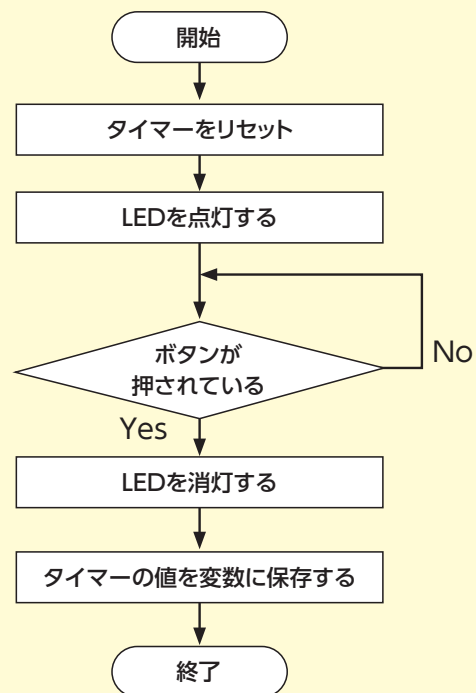
【課題】

メインユニットの LED が点灯してから、ボタンが押されるまでの時間を測定するタイマーを作成します。
ボタンを押すまでの時間を測定するためのタイマーなので、プログラムがスタートしてから LED が点灯するまでの時間（x 秒）はランダムに切り替わるようにしてください。



①フローチャートを考え、下の赤枠内に書き入れましょう。

解答例



基本

早押しボタンを作成しよう

年 組 番 名前

知識技能①



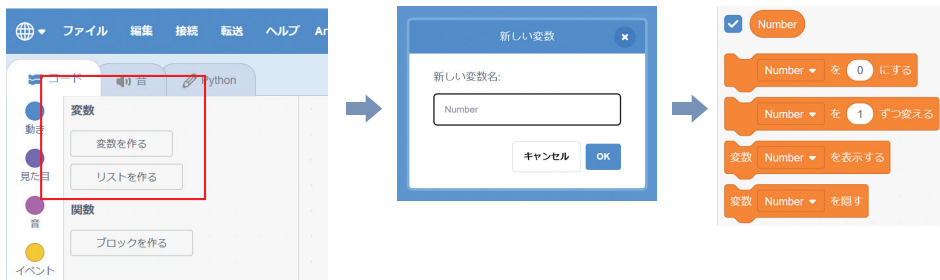
変数ブロック

変数は情報を保存するために使われます。その変数を使うためには、変数ブロックを使う必要があります。
変数ブロックに記入された情報が変数に保存されます。

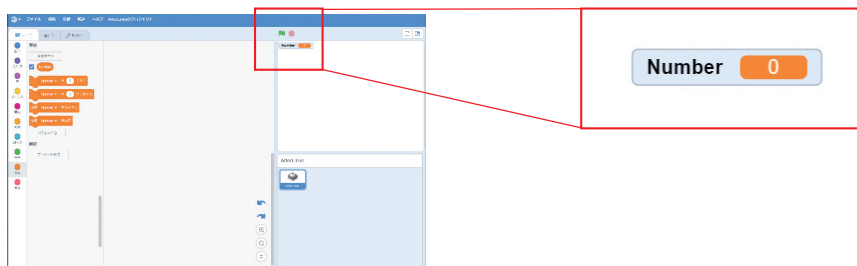
変数ブロックを表示させる

ソフトウェアを起動しただけでは、変数ブロックは表示されていません。そのため、まず変数ブロックを表示させます。

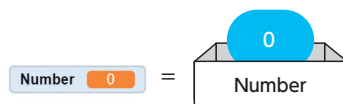
- (1) 変数 から 変数を作る をクリックします。
- (2) 変数の名前を設定します。この例では「Number」という名前を設定しています。この変数に保存させる情報に関連した名前だと、整理しやすくなります。
- (3) 「OK」をクリックすると、以下のブロックがパレットに表示されます。



- (4) 変数を作ると、右上の画面に以下の表示が追加されます。この表示から、現在この変数にどのような情報が保存されているのかを確認することができます。



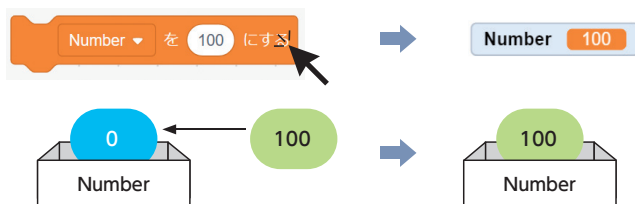
以下の表示の場合、「Number」という変数に「0」という情報が保存されていることを確認することができます。



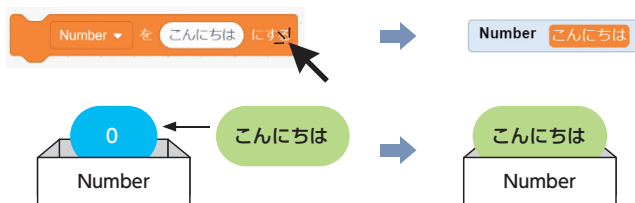
変数に情報を保存する

- (1) 変数に情報を保存するためには **Number ▾ を 0 にする** のブロックを使います。
- (2) ブロックをスクリプトエリアにドラッグ&ドロップします。
- (3) 青枠部分に保存したい情報を記入し、ブロックをクリックします。 **Number ▾ を 0 にする**
- (4) **Number 0** の表示部分が、記入した文字と同じになります。これで記入した情報が変数に保存されたこととなります。

例 変数「Number」に「100」を保存する場合

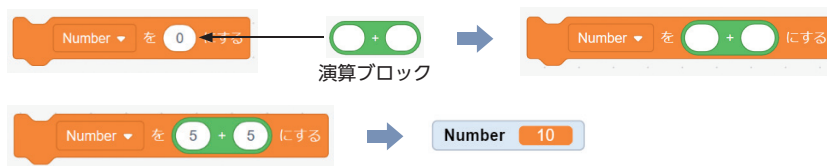


例 変数「Number」に「こんにちは」を保存する場合

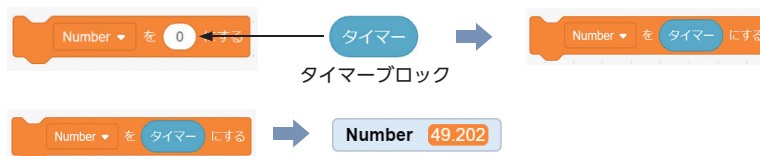


- (5) 保存したい情報を記入する部分には、**◯◯** の形のブロックを入れることができます。「演算ブロック」や「タイマーブロック」（タイマーブロックの詳細は P.29）などが当てはまります。

例 変数「Number」に「5 + 5」を保存する場合



例 変数「Number」に「タイマーの時間」を保存する場合



基本

早押しボタンを作成しよう

年 組 番 名前

知識技能①

変数ブロックを使って計算をする

- (1) 以下のようなプログラムが作成されていた場合、プログラムが実行されるたびに「Number」に保存される数はどのように変化するでしょうか。実際にプログラムを作成し、実行して、以下の表に当てはまる数字を記入しましょう。



プログラムを実行すると…

Number ?

プログラムを実行した回数	0 回	1 回	2 回	3 回	4 回
「Number」の値	0	6	12	18	24



タイマーブロック

時間を測定するためには、カテゴリ内  の「タイマーブロック」を使います。
調べる

タイマー

このブロックを使って時間を計測します。単位は秒です。タイマーはソフトウェアを起動したタイミングから自動的に計測を始めます。クリックすると計測された時間が表示されます。

タイマー

246.345

タイマーをリセット

このブロックを使うとタイマーの値を 0 秒に戻すことができます。

タイマーはソフトウェアを起動したタイミングから時間の計測を始めているため、タイマーで秒数を計測する場合は、計測をスタートさせたいタイミングで「タイマーをリセット」を行う必要があります。

例 プログラムが実行されてからメインユニットのボタンを押すまでの時間を変数「Time」に保存するプログラム




プログラムが実行されるごとにタイマーの値をリセットします。

ボタンが押された時のタイマーの値を、「Time」という変数に保存します。

知識技能①

！ 乱数ブロック

乱数（ランダムな数）を発生させるためには、カテゴリ内  の「乱数ブロック」を使います。
以下のブロックの場合、1～10の間でランダムに整数を出力します。

1 から 10 までの乱数

乱数ブロックをクリックしてみましょう。以下のようにクリックするたびに異なる数字が出力されます。

1 から 10 までの乱数

7

1 から 10 までの乱数

2

1 から 10 までの乱数

9

知識技能②

② P.26 で作成したフローチャートを参考にして、プログラムを作成しましょう。以下の「使用ブロック」を参考にしてください。

使用ブロック

LEDを  で点灯

メインユニットの LED を光らせるためのブロックです。

LEDを消灯

メインユニットの LED を消灯するためのブロックです。

1 秒待つ

指定した秒数だけプログラムをストップします。

本体のボタンが押されたとき

ボタンが押されたときに、このブロックの下にブロックをハットブロックといい、ブロックを接続します。

タイマー

時間を計測するためのブロックです。

タイマーをリセット

タイマーを 0 秒に戻すためのブロックです。タイマーは、このブロックで時間をカウントしています。

1 から 10 までの乱数

ランダムな数字を出力するためのブロックです。

time を 0 にする

変数の値を指定した値に変えるためのブロックです。

プログラミング
実習

解答例




基本

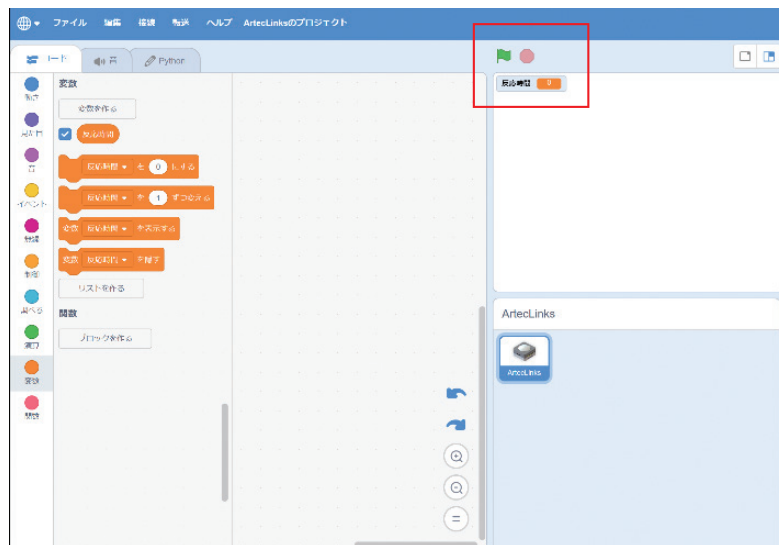
早押しボタンを作成しよう

年 組 番 名前

知識技能②

③プログラムが完成したら、ソフトウェア画面右上の  を押して、プログラムを実行しましょう。.....
メインユニットのLED が光ったらボタンを押しましょう。LED が光ってからボタンを押すまでの時間を計測しましょう。

計測された時間は以下の赤枠内の変数「反応時間」に表示されます。

プログラミング
実習

⑤達成できた項目にチェックを入れましょう。

- ☐ 「変数」について理解することができた
- ☐ 「乱数」について理解することができた
- ☐ 「変数」の使い方を理解することができた
- ☐ 「タイマーブロック」の使い方を理解することができた
- ☐ フローチャートを書くことができた
- ☐ 反応時間計測タイマーを作成することができた

基本

早押しボタンを作成しよう

年 組 番 名前

基本課題① 早押しボタンを作成しよう

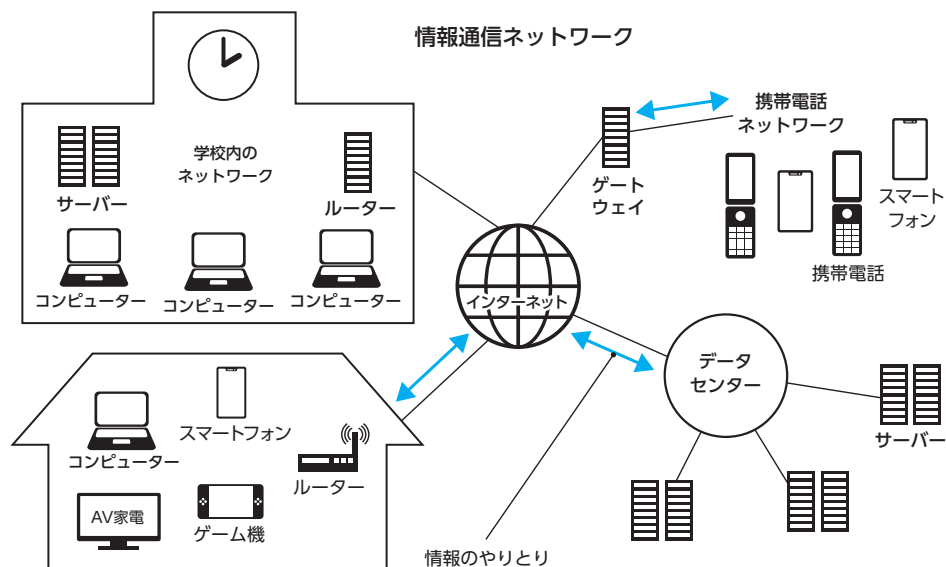
知識技能①



情報通信ネットワーク

「情報通信ネットワーク」とは、コンピューターやスマートフォンなどのデバイス同士がネットワークを使って、情報をやりとりする仕組みです。これは、手紙や電話と同じコミュニケーション手段の1つです。例えると、ネットワークは街中を走る道路のようなもので、データはその道路を通る車です。様々な目的地を行き交いながら情報をやり取りします。

ネットワークは、ルーターやゲートウェイ、データセンターなどによって構築されていて、電話線や Wi-Fi、光ファイバーなどの技術でつながっています。情報を迅速に伝えたり、ウェブサイトを閲覧したり、ビデオ通話を楽しんだりするのに役立っています。情報通信ネットワークは、現代のコミュニケーションと情報共有で中心的な役割を果たしています。

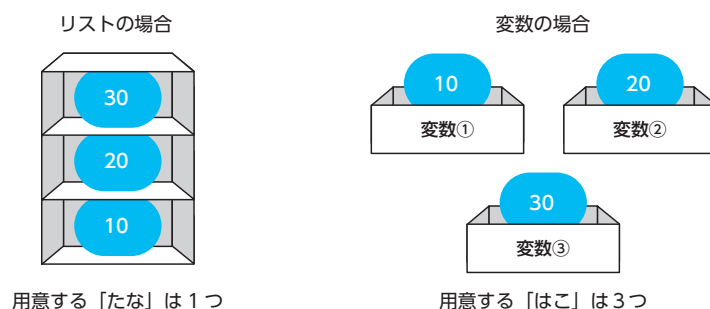


「リスト」とは、たくさんの情報を整理し、保存するために使われます。

変数では1つしか情報を保存することができませんでしたが、「リスト」では複数の情報を一括して保存することができます。

変数は「はこ」に例えることができたが、リストは「たな」に例えることができます。

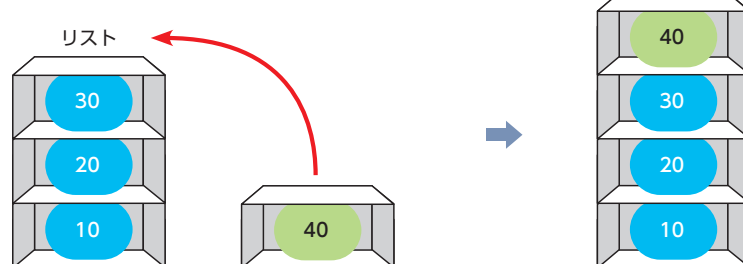
例 10, 20, 30 の3つの情報を保存する場合



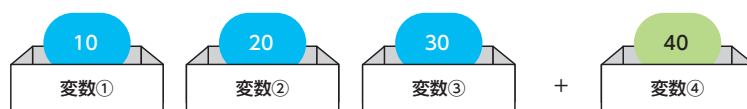
「リスト」は同じ種類のデータを一か所にまとめることができるので、データを効率的に管理することができます。また、リストはプログラムの中で情報を追加で保存したり、保存されている情報を消したりすることができます。

例 10, 20, 30 の3つの情報に「40」という情報を追加する場合

●リストの場合、「たな」を新たに用意する必要がない



●変数の場合、「はこ」を新たに用意する必要がある



基本

早押しボタンを作成しよう

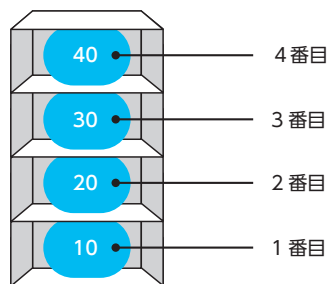
年 組 番 名前

知識技能①

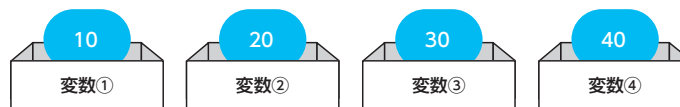
リストには、データが追加された順番通りに保存されます。そのため、早押しボタンのようにどちらのボタンが早く押されたのか判定する場合は、リストを使うと便利です。

例 10 → 20 → 30 → 40 の順番で情報が保存された場合

●リストの場合、保存された順番通りに「たな」に保存される



●変数の場合、保存された順番を知ることはできない



「アクティビティ図」とは、プログラムの概要とプログラムの流れを図解したものです。

特に複数のメインユニットで通信する場合、アクティビティ図を作成すると全体のプログラムの見通しを立てやすくなります。

通信を用いる場合は、プログラムが複雑になるため、混乱しないように事前にアクティビティ図を作成します。アクティビティ図でプログラム全体の概要を把握したのち、フローチャートを作成し、プログラムの詳細を決定していきます。

	フローチャート	アクティビティ図
特徴	プログラムの詳細を把握することができる。 このフローチャートを参考にして、プログラミングすることができる。	プログラム全体の動きを把握することができる。 複雑なプログラミングの際に混乱することなく、進めることができる。

アクティビティ図作成のルール

自動販売機の動作を例に、アクティビティ図を作成する流れを説明します。

(1) 登場するものを整理する

プログラミングの動作に関係するものを洗い出します。

例 自動販売機の動作に関係するもの



購入者



自動販売機

(2) 登場するものの動作を整理する

ここではフローチャートのように詳細な動作は必要なく、概要がわかれば十分です。



購入者

- お金をいれる
- 欲しい飲み物のボタンを押す
- 飲み物をとる



自動販売機

- 押されたボタンの飲み物を認識する
- 投入されたお金が飲み物の金額に足りているのか判断する
- 飲み物を出す

基本

早押しボタンを作成しよう

年 組 番 名前

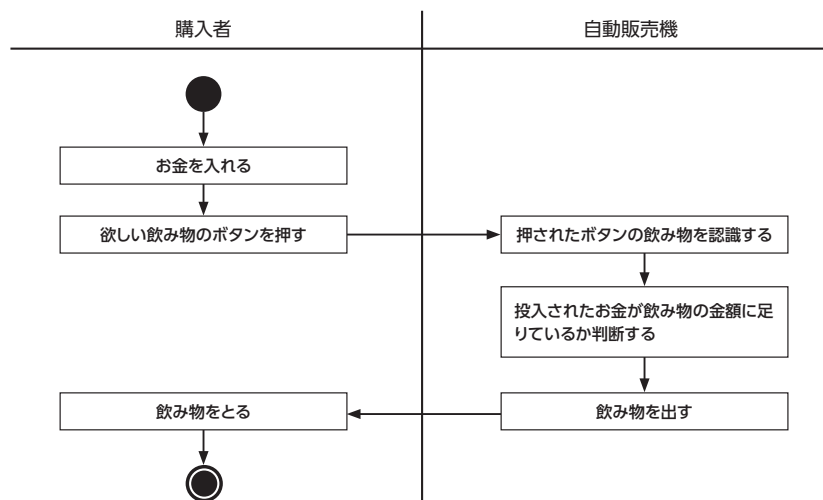
知識技能①

(3) 登場するモノに対応したレーンを書く

それぞれのレーンに対応した動作を順番に書いていきます。行われる順番に動作を矢印でつないでいきます。

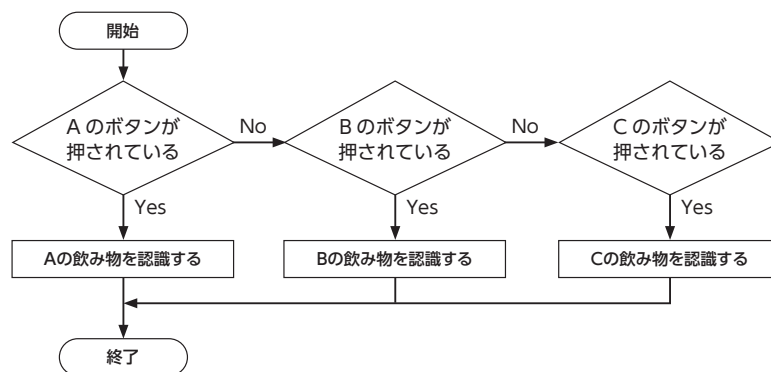
また、動作の最初には ● (開始ノード)、動作の最後には ● (終了ノード) の記号を付けます。

例 自動販売機のアクティビティ図



※アクティビティ図作成後、それぞれの動作についてフローチャートを作成し、詳細を決定します。

例 「押されたボタンの飲み物を認識する」という動作のフローチャート



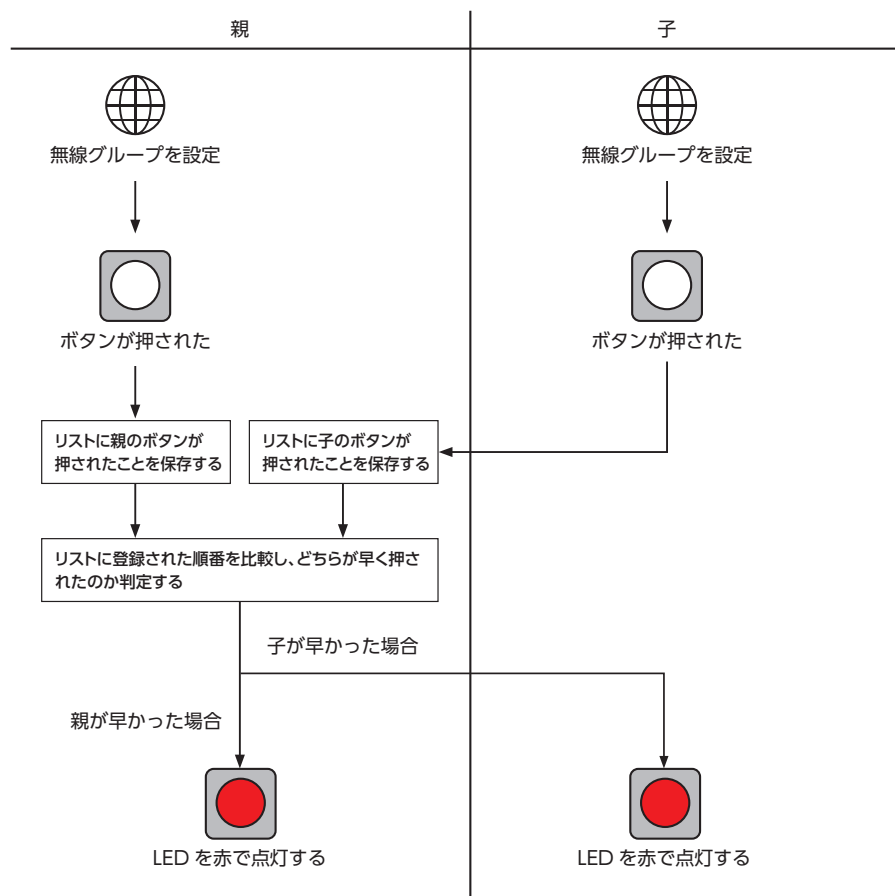
「早押しボタンシステム」をアクティビティ図に表しましょう

【課題】

次のような早押しボタンのプログラムを作成しましょう。

まず2人グループを組み、片方を親、もう片方を子として、親と子の役割分担をします。

親のメインユニットでは、親と子、どちらのボタンが早く押されているのかをリストを使って判定します。もし親のメインユニットのボタンが早く押されていた場合、自分のLEDライトを点灯します。子のメインユニットのボタンが早く押されていた場合、親から子に信号を送り、子のLEDライトを点灯させます。



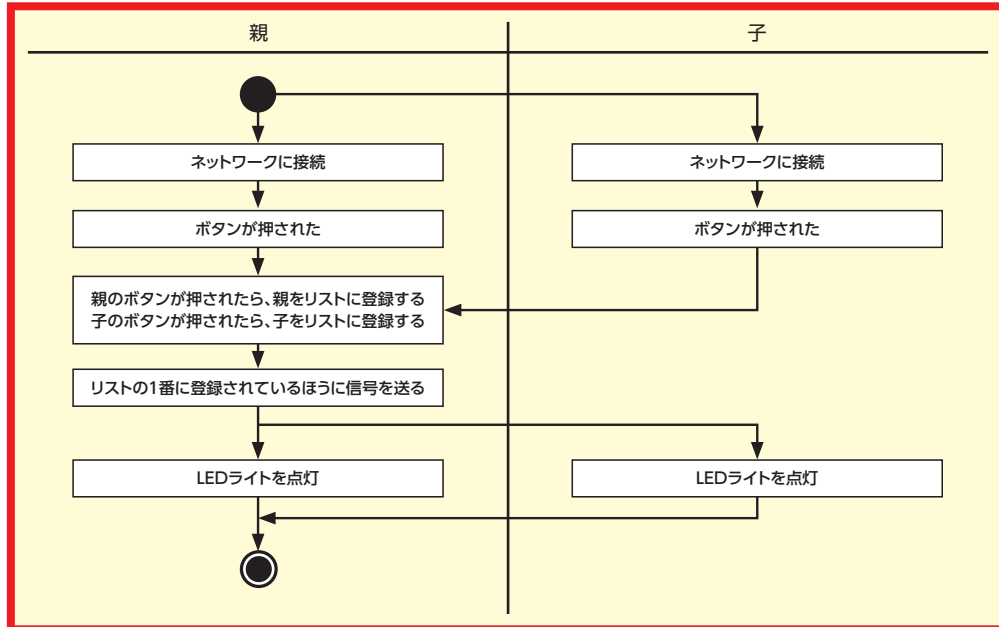
基本

早押しボタンを作成しよう

年 組 番 名前

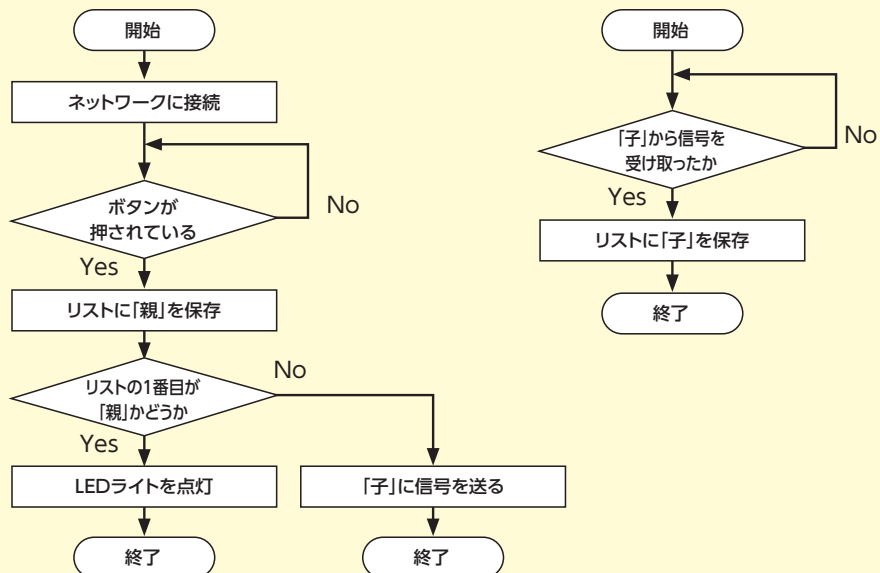
知識技能②

(1) アクティビティ図を考え、下の赤枠内に書き入れましょう。



(2) フローチャートを考え、下の赤枠内に書き入れましょう。親と子どちらを作成するか決めましょう。

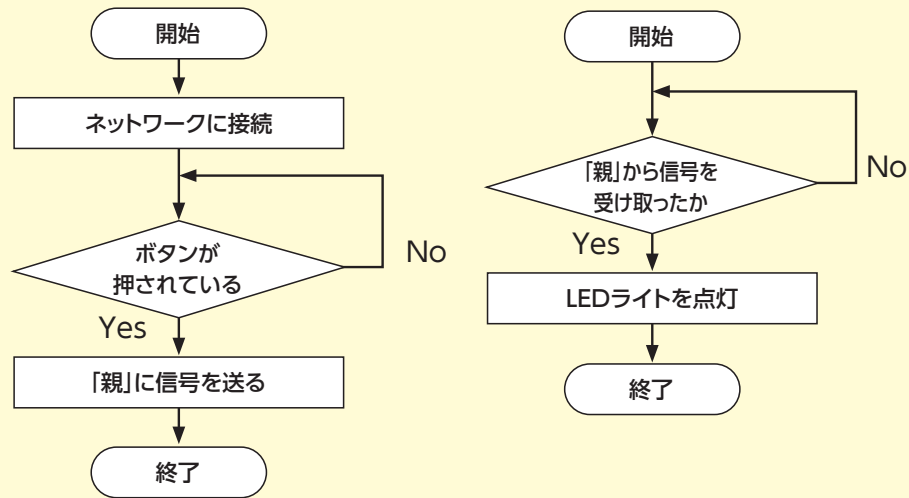
作成するプログラム（親の場合のフローチャート）




知識技能②

(2) フローチャートを考え、下の赤枠内に書き入れましょう。親と子どもどちらを作成するか決めましょう。

作成するプログラム（子の場合のフローチャート）



無線ブロック

メインユニットを通信させるためには、カテゴリ内  の「無線ブロック」を使います。
メインユニット同士を通信させるため、2人以上のグループを作成して行ってください。



無線グループを設定する


(1) ロボットモードになっていることを確認します。

〈モードの確認方法〉

- 画面左上のタブから「編集」を選択し、一番下に表示されるタブを確認します。
- ここが「キャラクターモードに切り替える」となっていれば、ロボットモードです。
ここが「ロボットモードに切り替える」となっていれば、キャラクターモードです。



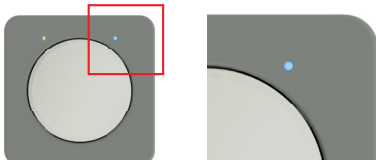
(2)  の  を使って、複数のメインユニットを同じグループのネットワークに接続して通信することができます。

 この数字は、無線グループを設定するための暗証番号のような役割を担います。そのため、同じグループに接続するメインユニット同士は同じ数字を入力するようにしてください。

(3) 以下のようにブロックを組み、 を押してプログラムを実行します。



(4) メインユニットが以下のように青く光っている場合、無線グループの設定は完了です。



基本

早押しボタンを作成しよう

年 組 番 名前

知識技能①

無線通信を行う

- (1) 無線のグループを設定 0 ブロックを使って、無線グループを設定します。

- (2) 送信側と受信側のプログラムを作成します。以下のブロックを使います。

無線で Hello を送信

無線で同じネットワークに接続しているメインユニットに信号を送信するためのブロックです。ブロックの「Hello」の部分は自由に設定することができます。

無線で Hello を受信したとき

無線で同じネットワークに接続しているメインユニットからの信号を受信するためのブロックです。信号を受信したとき、このブロックに続くプログラムが実行されます。

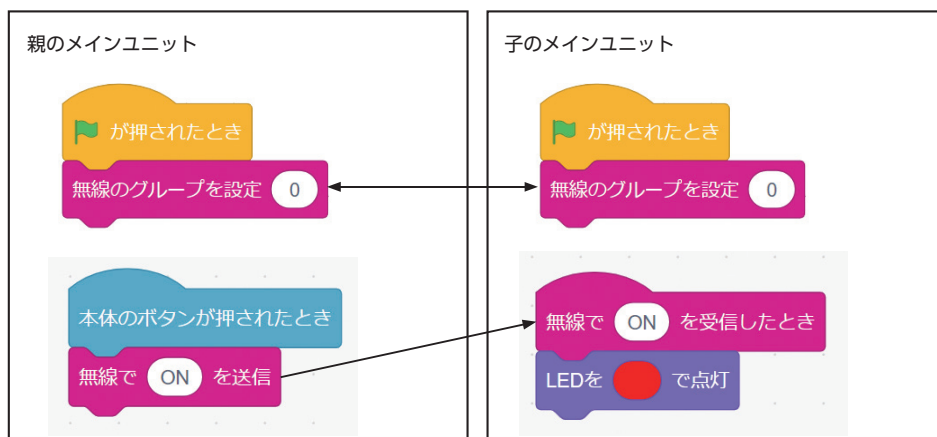
(送信側)

無線 の 無線で Hello を送信 ブロックを使って、相手（受信側）に信号（Hello という文字）を送信します。


(受信側)

無線 の 無線で Hello を受信したとき ブロックを使って、送信側からの信号を受信します。ブロックに記入された文字（イラストの場合は Hello）と同じ文字の信号を受信したとき、下に続くプログラムが実行されます。



例 親のメインユニットのボタンを押したときに、子のメインユニットのLEDライトを点灯するプログラム



！ リストブロック

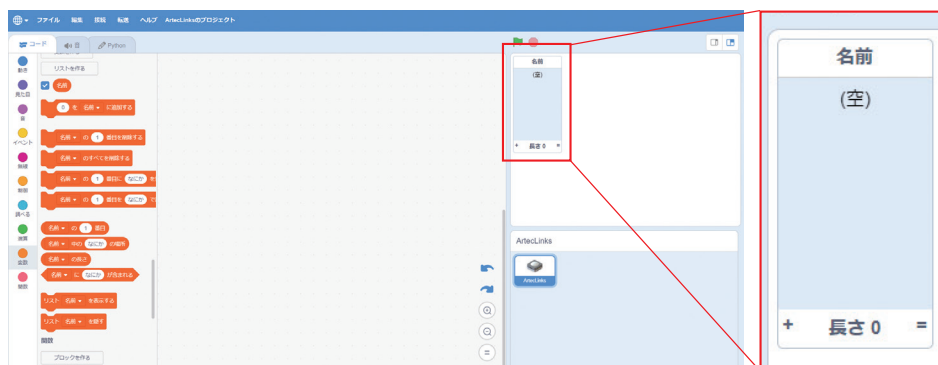
リストを使用するためには、カテゴリ内  の「リストブロック」を使います。

リストブロックを表示させる

- (1)  から  をクリックします。
- (2) リストの名前を設定します。この例では、「名前」という名前を設定しています。このリストに保存する情報に関連した名前にすると整理しやすくなります。
- (3) 「OK」をクリックすると、以下のブロックがパレットに表示されます。



- (4) また右上の画面には、以下の表示が追加されます。
この表示から、現在このリストにどのような情報が保存されているのかを確認することができます。
また、「長さ」の数字からリストに保存されている情報の個数を確認することができます。




基本

早押しボタンを作成しよう

年 組 番 名前

リストに情報を保存する


リストに情報を保存するために  を  に追加する のブロックを使います。

- (1) このブロックをスクリプトエリアにドラッグ&ドロップします。
- (2)  を 名前 ▼ に追加する の青枠部分に保存したい情報を記入します。
- (3) 情報を記入してプログラムを実行すると、以下のように情報がリストに保存されます。



リストに保存された情報を使う

リストの情報を使うために  の 1 番目 のブロックを使います。
リストからこのブロックの数字に対応した情報を呼び出すために使います。

- (1) このブロックをスクリプトエリアにドラッグ&ドロップします。
- (2)  の 1 番目 の青枠部分に呼び出したい情報の数字を入力します。入力後、このブロックをクリックすると、その情報が表示されます。

例 以下のようなリストの場合、次のように表示されます。

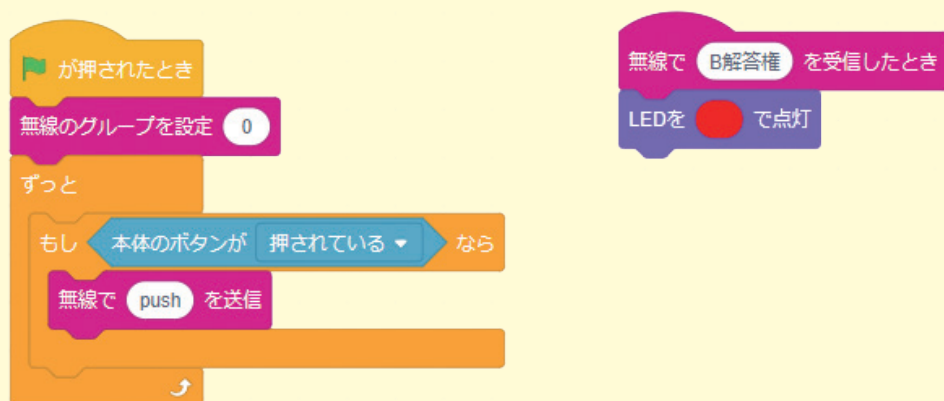


③ ②で作成したフローチャートを参考にして、プログラムを作成しましょう。以下の「使用 ……」ブロック」を参考にしてください。

作成するプログラム（親の場合）



作成するプログラム（子の場合）



基本

早押しボタンを作成しよう

年 組 番 名前

知識技能②

(子の場合)

使用ブロック

LEDを で点灯

メインユニットのLEDを光らせるためのブロックです。

無線のグループを設定 0

同じネットワークに接続するために、無線グループを設定するブロックです。

無線で Hello を送信

無線で信号を送信するためのブロックです。

無線で Hello を受信したとき

無線で信号を受信したときに、このブロックに続くプログラムを実行するためのブロックです。

ずっと

ずっと繰り返して実行するためのブロックです。このブロックの間に挟まれているブロックのプログラムをずっと繰り返し実行します。

もし なら

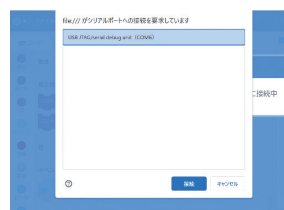
条件分岐を行うためのブロックです。条件が満たされたときに、このブロックの間に挟まれているブロックのプログラムを実行します。

⑤プログラムが完成したら、転送を行い、プログラムを実行しましょう。

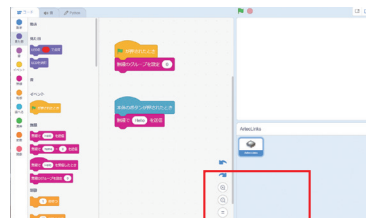
【転送方法】

基板間通信を行うためには、メインユニットに作成したプログラムを保存させる必要があります。この「プログラムを保存すること」を「転送」と言います。

- (1) プログラム作成後、タブから「転送」をクリックします。
- (2) 「USB JTAG…」を選択し、「接続」をクリックします。



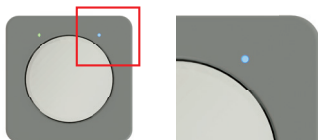
表示されていたユニットボードが消えたら、転送は完了です。



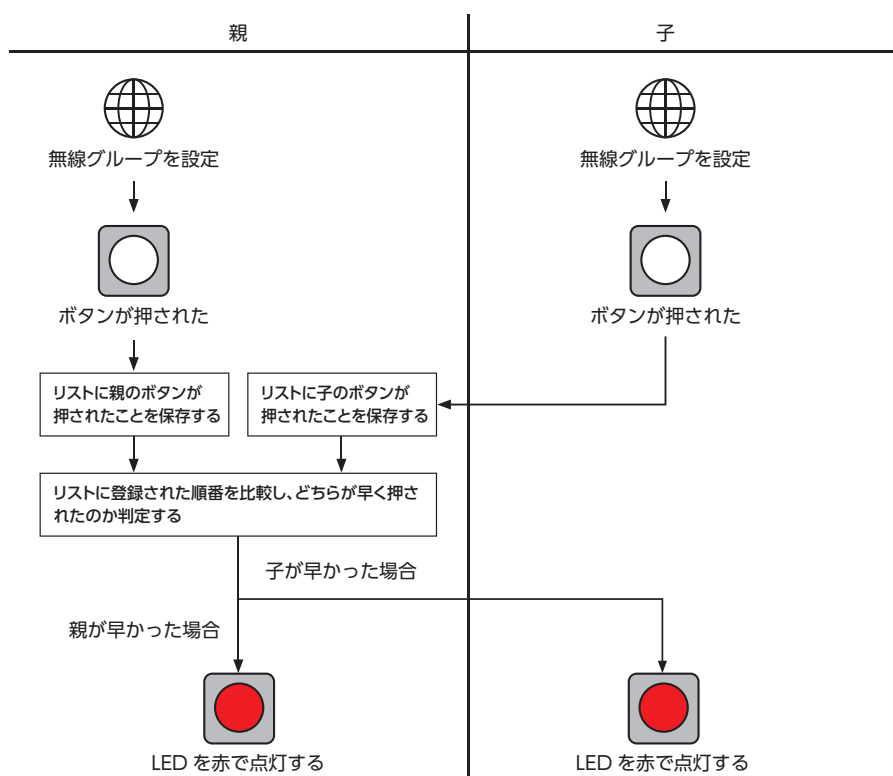
知識技能②

【転送後のプログラム実行方法】

- (1) 以下の画像のようにメインユニットに青いランプが点灯したことを確認し、無線グループが設定されたことを確認します。



- (2) 無線グループ設定後、親と子でメインユニットのボタンを押し、早く押したほうのボタンのみ光ることを確認しましょう。※再度、プログラムを実行したい場合は、親と子の電源を一度オフにして、(1)を行ってください。



⑥達成出来た項目にチェックを入れましょう。

- ☐ 「情報通信ネットワーク」について理解することができた
- ☐ 「リスト」について理解することができた
- ☐ 「アクティビティ図」について理解することができた
- ☐ 「無線通信ブロック」の使い方を理解することができた
- ☐ 「リストブロック」の使い方を理解することができた
- ☐ アクティビティ図を作成することができた
- ☐ 早押しボタンを作成することができた

基本

早押しボタンを作成しよう

年 組 番 名前

知識技能①

まとめ

「変数」と「リスト」どちらを使用すべきでしょうか。当てはまる方を下の赤枠内に入力してください。

- ① センサーで測定した気温を用いて、熱中症アラートを作成する場合

➡ 変数

- ② 20 人分の数学のテストの平均値を計算するプログラムを作成する場合

➡ リスト

- ③ 買い物の合計金額を自動で計算するプログラムを作成する場合

➡ リスト

フローチャート・アクティビティ図、それぞれの特徴を以下の表に入力してください。

	フローチャート	アクティビティ図
特徴	<p>プログラムの詳細を把握することができる。</p> <p>このフローチャートを参考にして、プログラミングすることができる。</p>	<p>プログラム全体の動きを把握することができる。</p> <p>複雑なプログラミングの際に混乱することなく、進めることができる</p>



基本セット 教員用

テキストに関するお問い合わせ

株式会社 **アーテック** お客様相談窓口



◀Webからのお問い合わせはこちら
<https://www.artec-kk.co.jp/contact/>

お電話でのお問い合わせはこちら
TEL 072-990-5656

B095762 K0325