ジュニア・プログラミング検定

Bronze（3級）対策講座

Scratch部門｜小中学生向け

目次

講座の全体構成を3列で整理

ジュニア・プログラミング検定とは？

Scratchで「指示を満たす作品」を時間内に完成させる実技試験。創造性も評価します。

Scratchで作品を完成

ゲームや物語など、問題文と完成例に沿って1つの作品を作ります。

レベル別に評価

Entry／Bronze／Silver／Goldで、ブロックの役割と活用力を段階的に測定。

アレンジ問題で創意工夫

必須要件を満たした上で、自分の工夫を加え、その意図を説明します。

上位級は仕様変更も

Silver／Goldでは、既存のプログラムを理解・修正する力も問われます。

ポイント：必須要件の達成が最優先。動作確認と保存をこまめに行い、アレンジは1〜2点を丁寧に。

4つのレベル

Entry / Bronze / Silver / Gold を横並びで比較

Entry（4級）

基本操作と簡単な連動

条件分岐・繰り返しの入門

1～2スプライトで制作

Bronze（3級）

条件分岐・繰り返しで連動

変数・演算でルール作成

少数スプライトを制御

Silver（2級）

複数条件・入れ子構造

連動の複雑化に対応

仕様変更の理解・修正

Gold（1級）

高度な連動と表現力

複数スプライトを高度制御

仕様変更に柔軟対応

注：各級はScratchブロックの理解と活用力を段階的に評価します。

Bronze（3級）の認定基準

Scratch部門｜3級 Bronze

求められるスキルを4つのポイントで確認しよう

1

論理的に考えられる

単純な条件や筋道を用いて、手順立てて考えられる。

2

条件分岐・繰り返しが使える

「もし〜なら」「〜まで/回くり返す」などの制御を活用できる。

3

少数のスプライトを連動

メッセージなどでスプライト同士を連携させて動かせる。

4

変数と演算を扱える

スコアやラップ数などを変数で管理し、比較や加算ができる。

メモ：必須要件の達成を最優先。初期化・保存・動作確認をこまめに行いましょう。

試験概要

Bronze（3級） | Scratch部門

試験時間
40分

出題形式
Scratch実技（指示を満たす作品を制作）

合格基準
得点率60%以上

受験料
2,800円（税込）

補足：対応バージョンは Scratch 2.0／3.0。出題例として「レースゲーム」＋アレンジ問題があります。

出典：サーティファイ「ジュニア・プログラミング検定」公式サイト

出題範囲① 使用ブロック（動き・見た目）

キャラを自然に動かし、見た目を切り替える基本ブロックを確認

動き（Motion）

主な用途

座標や向きを使ってキャラを移動させる

画面端の処理や回転方法を調整する

ブロック例

10歩動かす　x座標を( )にする　x座標を( )ずつ変える

( )度に向ける　端に触れたら跳ね返る

コツ：向き＋座標を組み合わせると、操作性のよい動きになります。

見た目（Looks）

主な用途

コスチューム切替でアニメーション表現

大きさや効果で演出、メッセージ表示

ブロック例

次のコスチュームにする　コスチュームを( )にする

大きさを( )%にする　色の効果を( )ずつ変える

( )と言う [2] 秒

コツ：動きに合わせてコスチューム切替を入れると滑らかに見えます。

ポイント：初期化（向き・位置・コスチューム）→ 操作入力 → 動き と見た目 をセットで設計すると、作品が安定して仕上がります。

出題範囲② 使用ブロック（音・制御）

演出とロジックをつくる基本ブロックをカードで確認

音（Sound）

主な用途

- 効果音やBGMで動きにあわせて演出する
- 音量を調整して聴きやすくする

ブロック例

(音) の音を鳴らす　すべての音を止める　音量を ( ) % にする　音量を ( ) ずつ変える

コツ：動きやイベントに合わせて効果音を鳴らすと、操作感がアップします。

制御（Control）

主な用途

- 処理の順番や回数・条件を決める
- ゲームの進行（ループ・待ち）を作る

ブロック例

ずっと　( ) 回繰り返す　( ) 秒待つ　もし (条件) なら　もし〜なら〜でなければ

コツ：初期化 → ずっと → 条件分岐の順で組むと、安定して動作します。

ポイント：音は演出、制御はロジック。両方を組み合わせて、わかりやすく遊びやすい作品に仕上げましょう。

出題範囲③ 使用ブロック（調べる・演算・変数）

当たり判定やルール作り、スコア管理の要となる3カテゴリを確認

調べる（Sensing）

主な用途

キー入力やマウス位置、接触の判定

ゴール・障害物などの当たり判定

ブロック例

キー( )が押された？　　( )に触れた？　　色( )に触れた？

マウスのx座標

コツ：当たり判定は色 or スプライト名に統一するとデバッグが楽。

√× 演算（Operators）

主な用途

数の計算、比較や論理式で条件づくり

乱数でゲーム性（運要素）を追加

ブロック例

( ) + ( )　　( ) > ( )　　( ) = ( )　　( ) かつ ( )　　乱数(1)から(10)

コツ：比較＋論理を条件分岐に入れて、正確な判定を作る。

変数（Variables）

主な用途

スコア・タイム・ラップ数などの記録

開始時の初期化と表示・非表示の制御

学習内容① Scratchの基本操作

基本UIとスクリプトの実行、スプライト同士の連動をおさえよう

基本の操作

- ステージ・スプライト・バックドロップの役割を理解
- スクリプトをドラッグ＆ドロップし、緑の旗で実行
- イベント（例：キーが押されたとき）で操作を受け付ける
- こまめに保存、作品名とコメントで変更点を記録

連動の基本

- メッセージ送受信でスプライト同士を連携
- 初期化 → ずっと → 条件分岐の順で安定動作

スプライト連動ミニ図

図のように、送る→受け取るの流れを作ると、複数スプライトが同時に動きやすくなります。

メモ：開始時に位置・向き・変数をまとめて初期化し、実行ごとの動作を安定させましょう。

学習内容② 条件分岐と繰り返し

「もし〜なら」とループを組み合わせて、安定して動くゲームの土台を作ろう

条件分岐（もし〜なら）

「もし〜なら」「もし〜なら〜でなければ」で分かれ道を作る

比較や当たり判定と組み合わせてルール化（例：ゴールに触れたらクリア）

繰り返し（ループ）

メイン処理はずっとで常に監視・更新

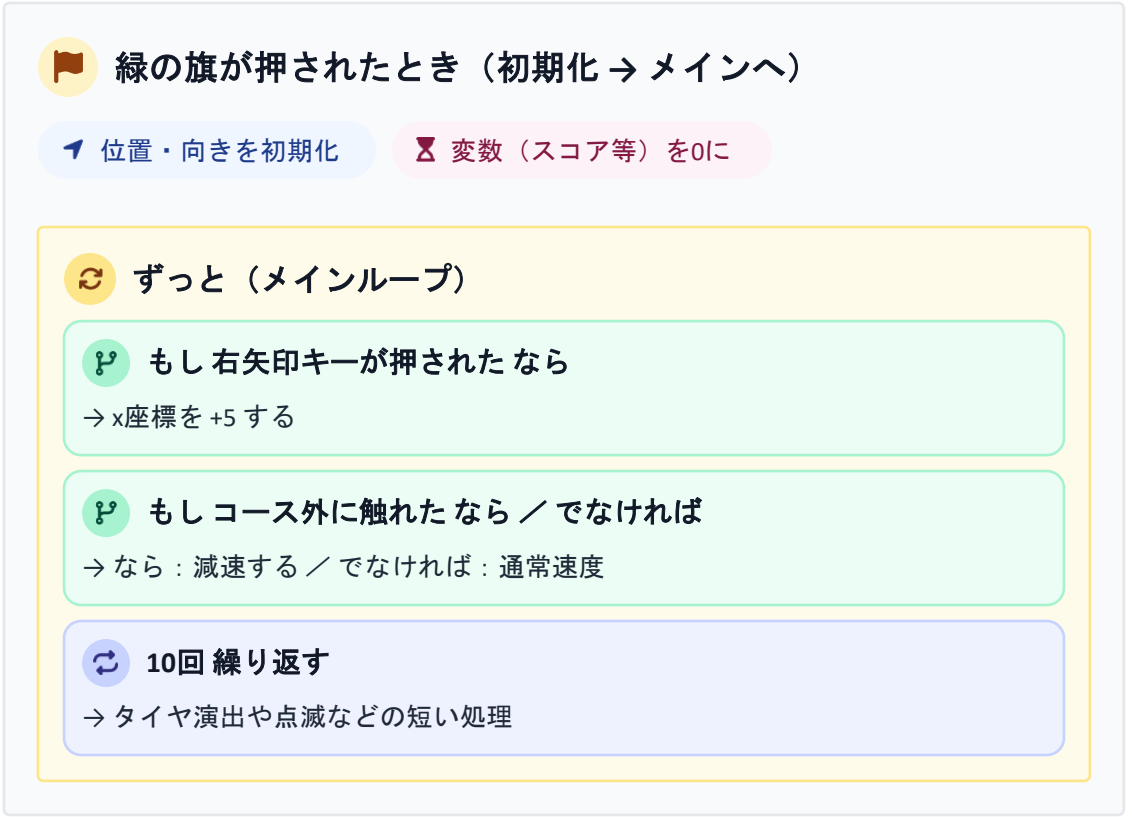
回数が決まるなら◯回繰り返す、条件なら〜まで繰り返す

組み合わせのコツ

初期化 → ずっと → もし〜ならの順で安定

よくあるミス：無限ループ（条件が変わらない）／条件の取りこぼし

ブロックの組み合わせ例



「ずっと」の中で入力をチェックして条件分岐するのが基本。回数ループは短い演出に使うと整理しやすいです。

ヒント：〜まで繰り返すでは、条件がいつ変わるかを必ず用意（カウンタ加算・位置更新など）して無限ループを防ごう。

学習内容③ 変数と演算

変数の作成・初期化、スコア管理、演算ブロックの使い方をおさえよう

変数の作成と初期化

- メニュー「変数」→変数を作るで作成（例：スコア）
- ゲーム開始時に0にする（初期化）を忘れない
- 必要に応じて表示オン/オフと表示位置を調整

スコア管理の基本

- イベントに合わせて1ずつ変えるや-1ずつ変える
- タイム/ラップ数なども変数で管理できる

演算ブロックの使い方

- 比較（>, <, =）で勝敗やクリア条件を判定
- 論理（かつ／または）で条件を組み合わせる

図解：初期化 → 加算 → 比較 → 判定

ヒント：スコア表示は見やすい位置に配置。開始時にまとめて初期化すると安定します。

メモ：初期化 → 加算 → 比較 → 判定の順で考えると、バグを見つけやすくなります。

学習内容④ 座標の理解

x座標・y座標・向き／画面端判定／スタート位置の初期化をおさえよう

座標と向きの基礎

ステージの中心は (0, 0)。x は左右、y は上下

向きの角度は 0°(右), 90°(上), 180°(左), -90°(下)

移動は「xを◯ずつ変える」「yを◯ずつ変える」でコントロール

画面端と初期化

端に触れたら「跳ね返る」で折り返し

開始時に 位置・向き・変数 をまとめて初期化

コース外に出たら スタート位置へ戻す 処理を入れる

コツ

移動量は小さめにし、毎フレーム確認すると安定

ゴールや外周の判定は、色やスプライト接触で簡潔に

座標と向きのミニ図

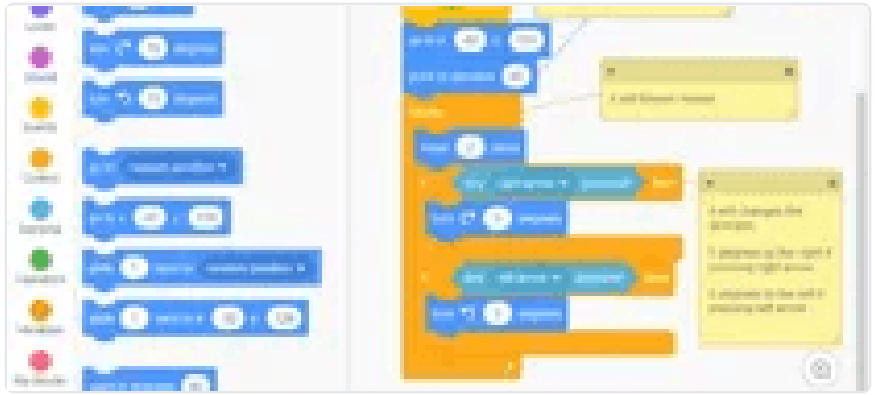
図のように、座標と向きを意識して「初期化 → 移動 → 端で処理」を作ると安定します。

メモ：レースゲームではスタート時に座標と向きをそろえ、毎周回で位置を確認するとズレを防げます。

サンプル問題：レースゲームを作ろう

矢印キーで操作し、周回してゴール。5ステップで完成させよう

完成イメージ



入力：矢印キー

目的：3周でゴール

画像は参考イメージです。

作り方（5ステップ）

1 コースとスタート位置を配置。変数「ラップ」を作り、開始時に0にする。

2 矢印キーで前進・回転。向きに合わせて進む。

3 コース外に触れたら減速、必要ならスタートへ戻す処理。

4 スタートライン通過を検知してラップを+1。重複加算防止のフラグを用意。

5 ラップ=3でゴール演出。アレンジを1つ加え、内容を説明。

提出チェック：初期化（位置・向き・変数）／当たり判定（色 or スプライト）／指示どおり＋アレンジ説明

対策方法と学習スケジュール

週2回 × 4週間のモデル（合計8セッション）

W1

W2

Week 1

A: 基本操作・UI

B: 動き・見た目復習

Week 2

A: 条件分岐・繰り返し

B: 当たり判定（色/接触）

Week 3

A: 変数・演算の基礎

B: スコア/ラップ実装

Week 4

A: 模擬制作（レース）

B: アレンジ練習＋本番リハ

コツ：指示を満たす → こまめに保存 → テスト → アレンジは1点を丁寧に

合格のポイント

必須要件・初期化・当たり判定・変数・アレンジ・動作確認の6項目をチェック

必須要件
問題の指示をすべて満たすことを最優先。

初期化
開始時に位置・向き・変数をまとめて設定。

当たり判定
色やスプライト接触など単純で確実な方法。

変数
名前と初期値を明確にし、更新は整理。

アレンジ
1〜2点を丁寧に。工夫の意図を説明できる。

動作確認
こまめに保存→テスト→修正を繰り返す。

提出前セルフチェック：上の6項目にすべて✓が付くか確認しよう。

学習リソース

公式サイト／サンプル問題／公式テキスト／Scratch公式／模擬問題集をカードで紹介

公式サイト（検定）

サーティファイ「ジュニア・プログラミング検定」の最新情報。

サイトへ

サンプル問題

出題イメージと完成例ムービーで要件を確認。

見る

公式テキスト（FOM）

Scratch 3.0対応の公認教材。全級に対応。

Scratch公式

エディタ／チュートリアルで基本操作を学習。

模擬問題集（例）

Bronze対策の練習問題で本番を想定。

参考

注記：外部サイトへ移動します。内容・価格等は各サイトで最新情報をご確認ください。

まとめ・応援メッセージ

Bronze（3級）の要点と、合格へ向けたメッセージ

Bronze（3級）まとめ

- 条件分岐・繰り返し・変数でシンプルなゲームを完成
- 少数スプライトの連動と初期化がポイント
- 「指示どおり」＋アレンジ1〜2点（意図を説明）
- 合格ラインは得点率60%以上

応援メッセージ

小さな完成を積み重ねれば大丈夫。初期化 → テスト → 改善をくり返して、本番は自信をもって挑もう！

ポイント：「指示を満たす」ことを最優先に、アレンジは1点を丁寧に。こまめに保存・動作確認！