

D-2

## 情報Iのプログラミング教育に生成AIを利用することの是非

◆ 井 手 広 康 ◆

愛知県立旭丘高等学校 教諭

k619154u@gmail.com



## 本研究の目的

- 令和4年11月にOpenAI社から**生成AI**であるChatGPTが公表されると、急速に世界中に普及し、教育の場での利活用も広まっている
- 文部科学省のガイドラインでは、「**プログラミングの授業において、児童生徒のアイデアを実現するためのプログラムの制作に活用する**」との記述がある
- プログラミング教育における生成AIの**活用事例**はまだまだ少なく、授業での生成AIの活用を躊躇している教員は多い（はず）

プログラミング教育に**生成AI**を活用し、その**効果・影響**について分析する



# プログラミングの授業概要

## 対象生徒

 年生 **199名**（井手が担当する5クラス） ※残り4クラスは別の教員が担当

## 実施科目

**課題研究（情報）** ※学校設定科目（情報の代替科目）

## 実施時期

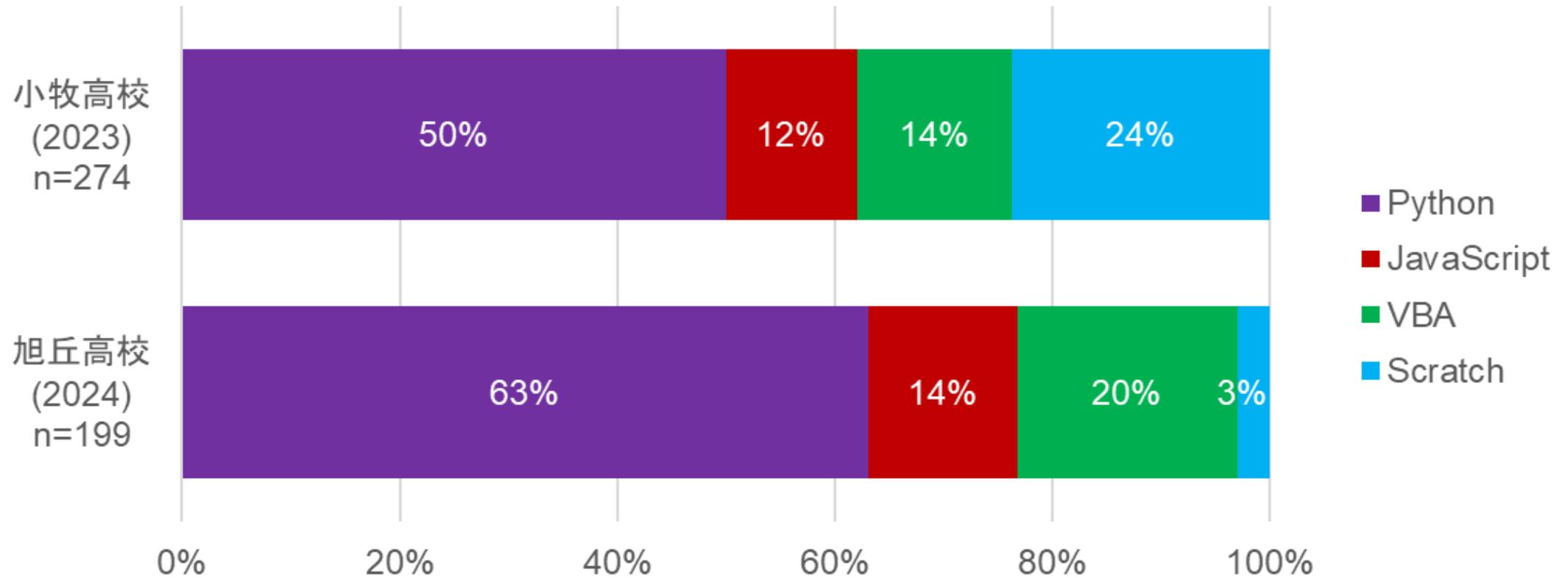
2024年9月～2025年1月（**1単位**）

## プログラミング言語

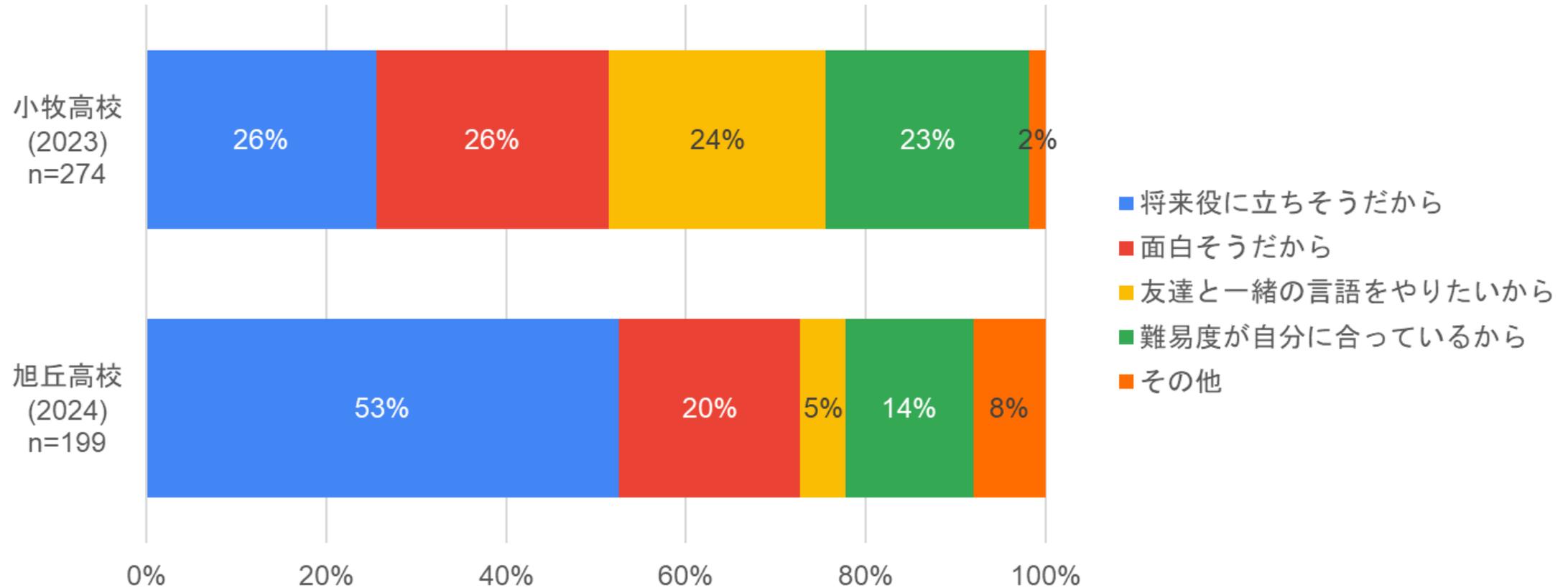
「**Python, JavaScript, VBA, Scratch**, その他」から各自で選択する



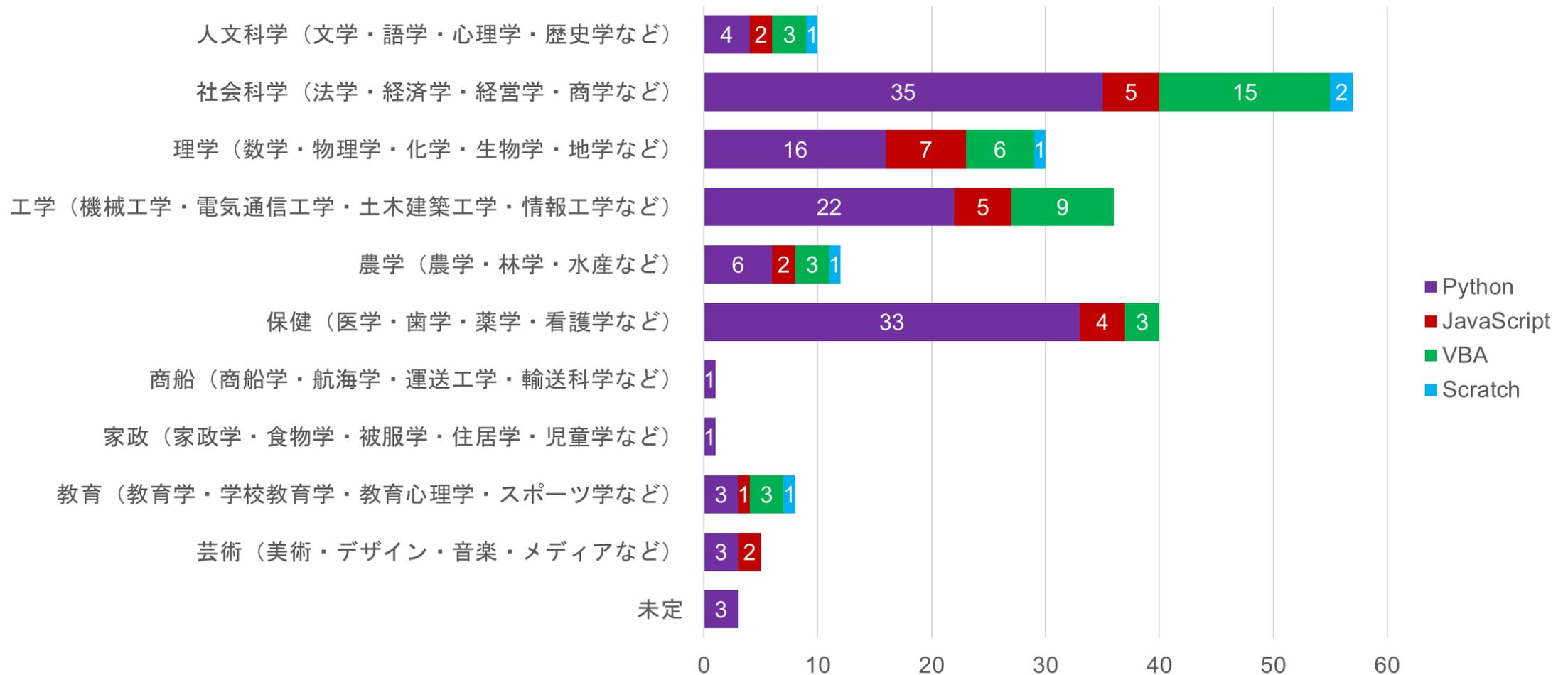
## 生徒が選択したプログラミング言語の内訳



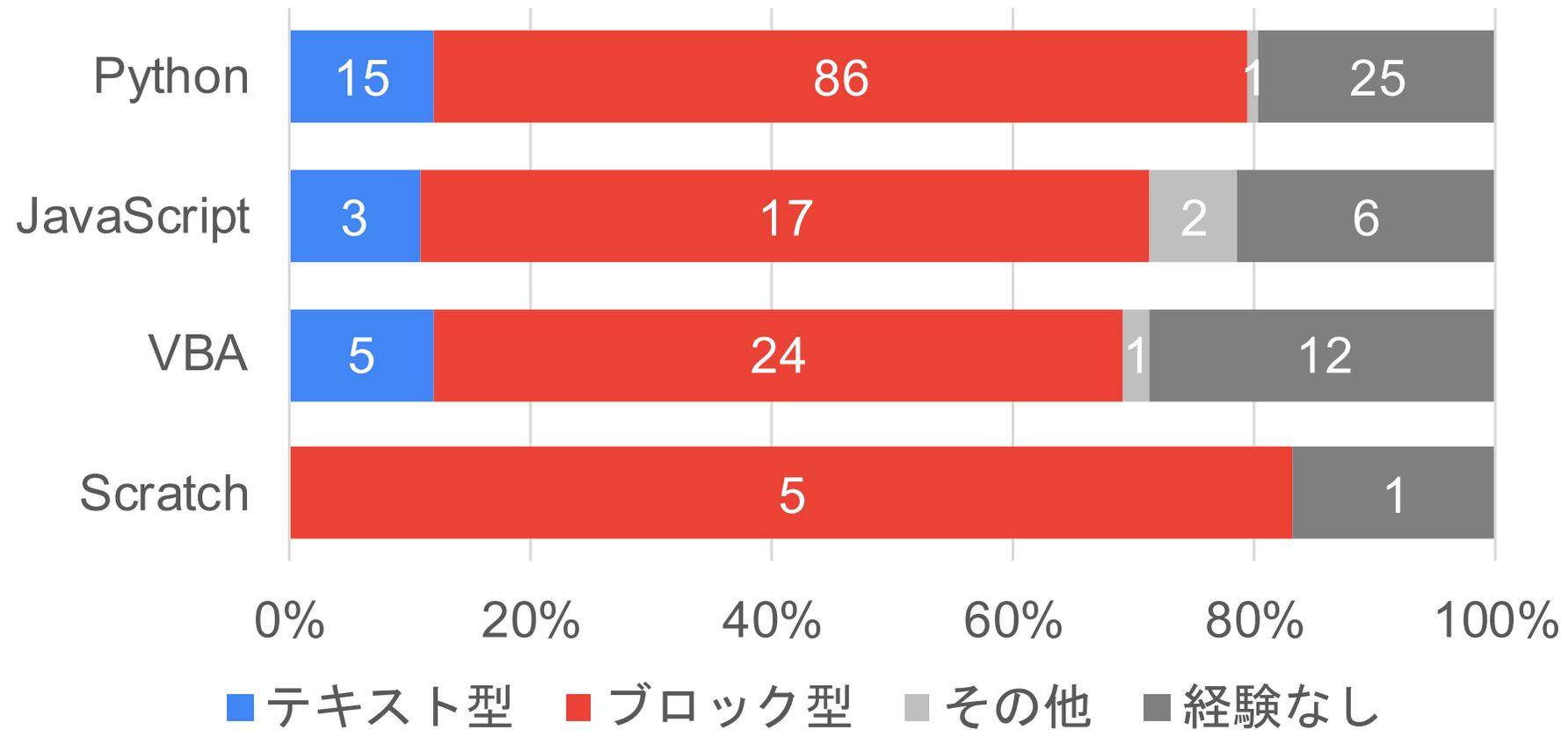
## プログラミング言語を選択した主な理由



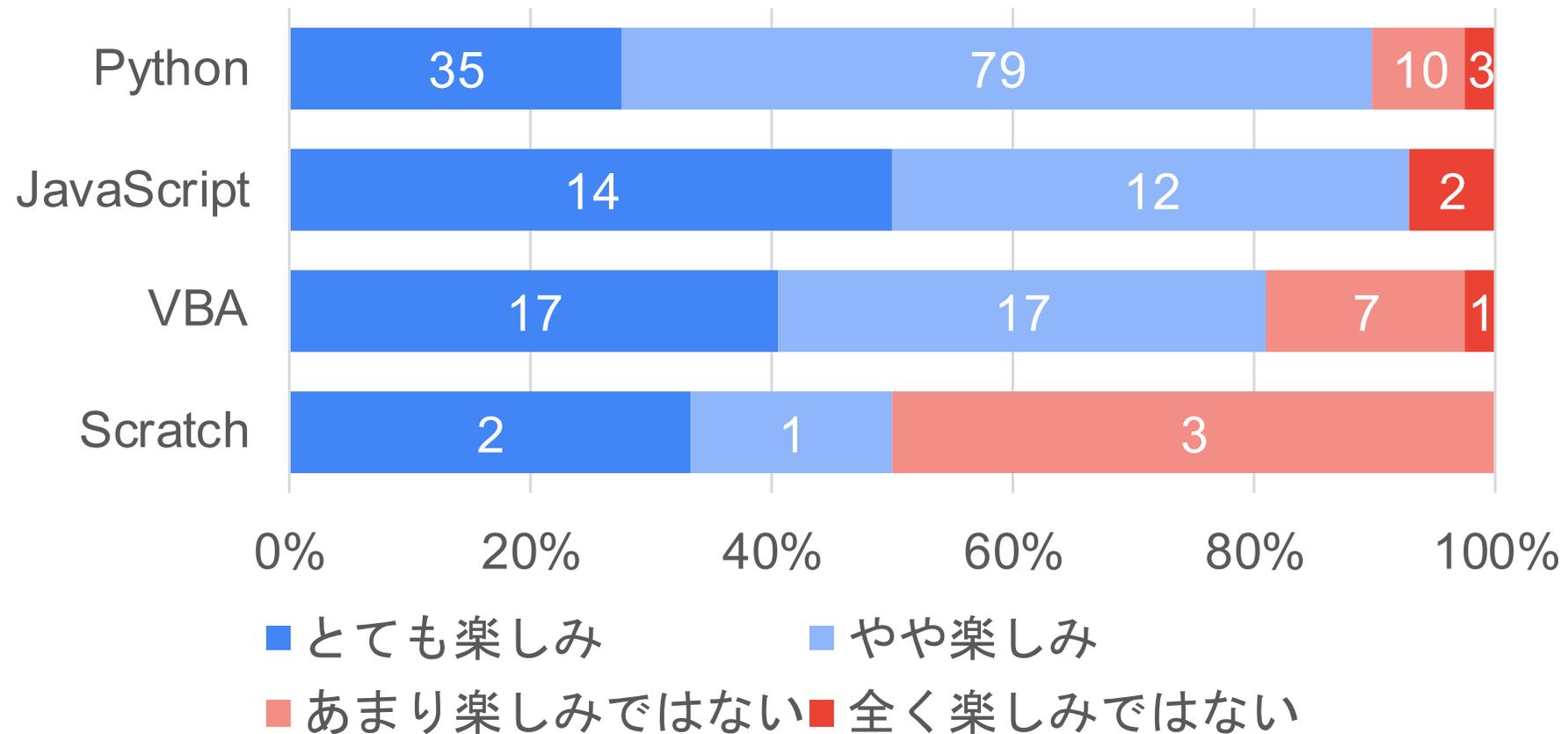
# プログラミング言語と進路希望



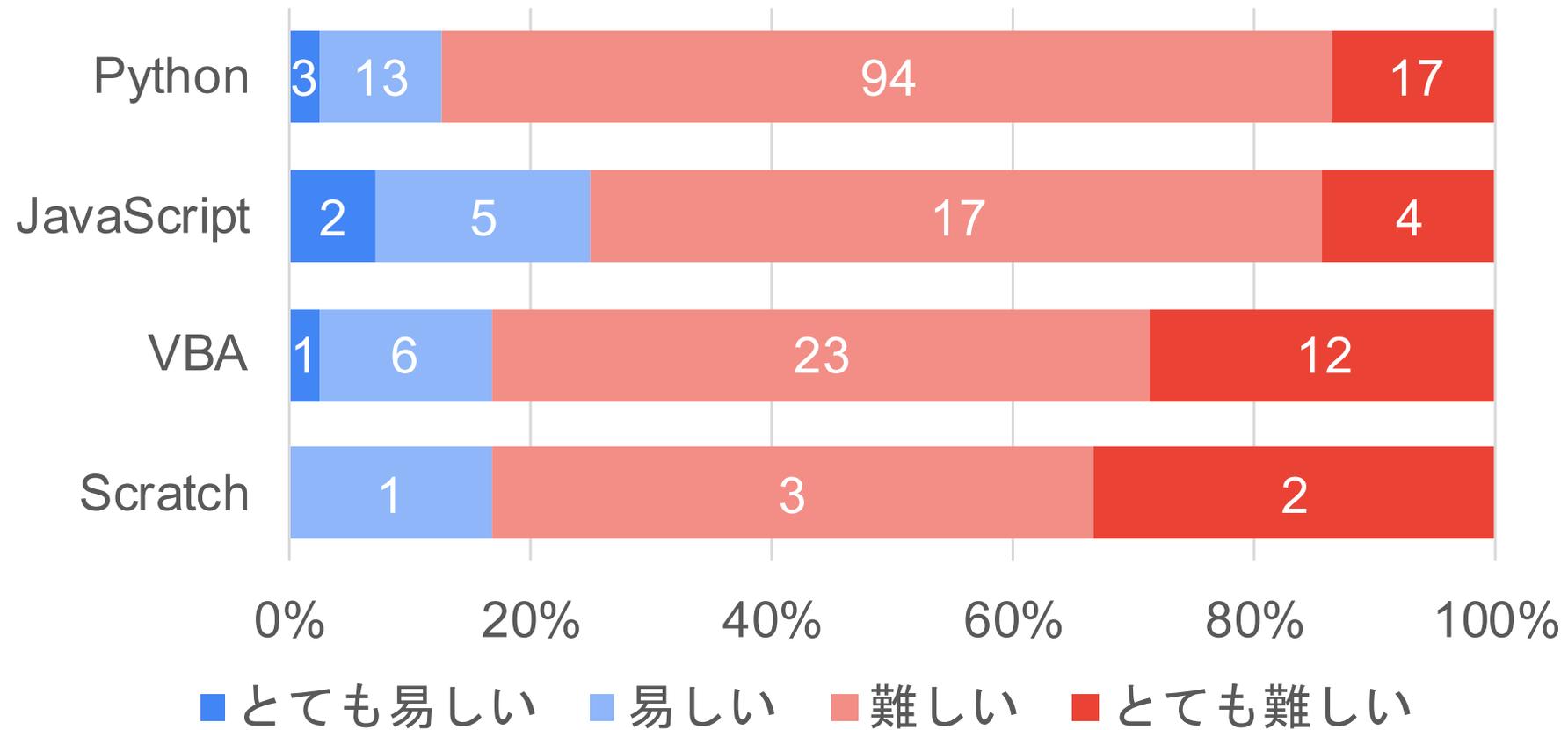
## 中学校でのプログラミング経験



## プログラミングに対する楽しさ



## プログラミングに対する難しさ



## プログラミングの授業内容

### 【前半】プログラミングの基礎

	メッセージ（画面出力）
	変数
	条件分岐（if文）
	繰り返し（for文）
	繰り返し（while文）
	配列／リスト（基礎）
	配列／リスト（応用）
	関数

### 【後半】プログラム制作と発表会

	プログラム制作①
	プログラム制作②
	プログラム制作③
	<b>中間レビュー会</b>
	プログラムの評価・改善
	<b>プログラム発表会</b>

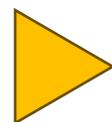


# 【前半】の基本的な授業の流れ

Step 

## 全体説明

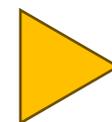
(約10分)



Step 

## 例題演習

(約20分)



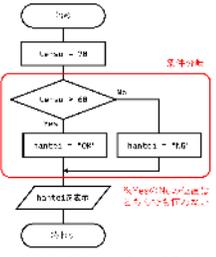
Step 

## 発展課題

(約20分)

**条件分岐**  
条件分岐は、条件式によって処理を変えるアルゴリズム(流れ)のこと。

Python	共通テスト用プログラム表記
<pre> tensu = 70 if tensu &gt; 60:     hantei = "OK" else:     hantei = "NG" print(hantei)                     </pre>	<pre> tensu = 70 もし tensu &gt; 60 ならば     hantei = "OK" そうでなければ     hantei = "NG" 表示する(hantei)                     </pre>



例：条件によってOKかNGを表示するプログラム

Python-Day3 条件分岐

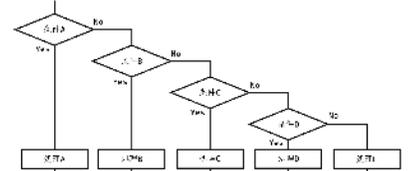
**例題**  
次のプログラムを実行し、結果を表示させなさい。

	Python	共通テスト用プログラム表記	備考
例3	<pre> ten = 90 if ten &gt;= 80:     print("Great") elif ten &gt;= 60:     print("Good") else:     print("Not Good")                     </pre>	<pre> ten = 90 もし ten &gt;= 80 ならば     表示する("Great") そうでなければ     もし ten &gt;= 60 ならば         表示する("Good") そうでなければ     表示する("Not Good")                     </pre>	5つの分岐 >=, >, <, <=, >=

※ elif を使用して複数の条件式を追加できる(いくつでも追加が可能)

Python-Day3 条件分岐

**発展**  
5つ以上の分岐があるプログラムを作成しなさい(条件式は何でも構いません)。また、変数をさまざまな値に変更して、それぞれの分岐が処理が正しいか、プログラムを実行して確認しなさい。

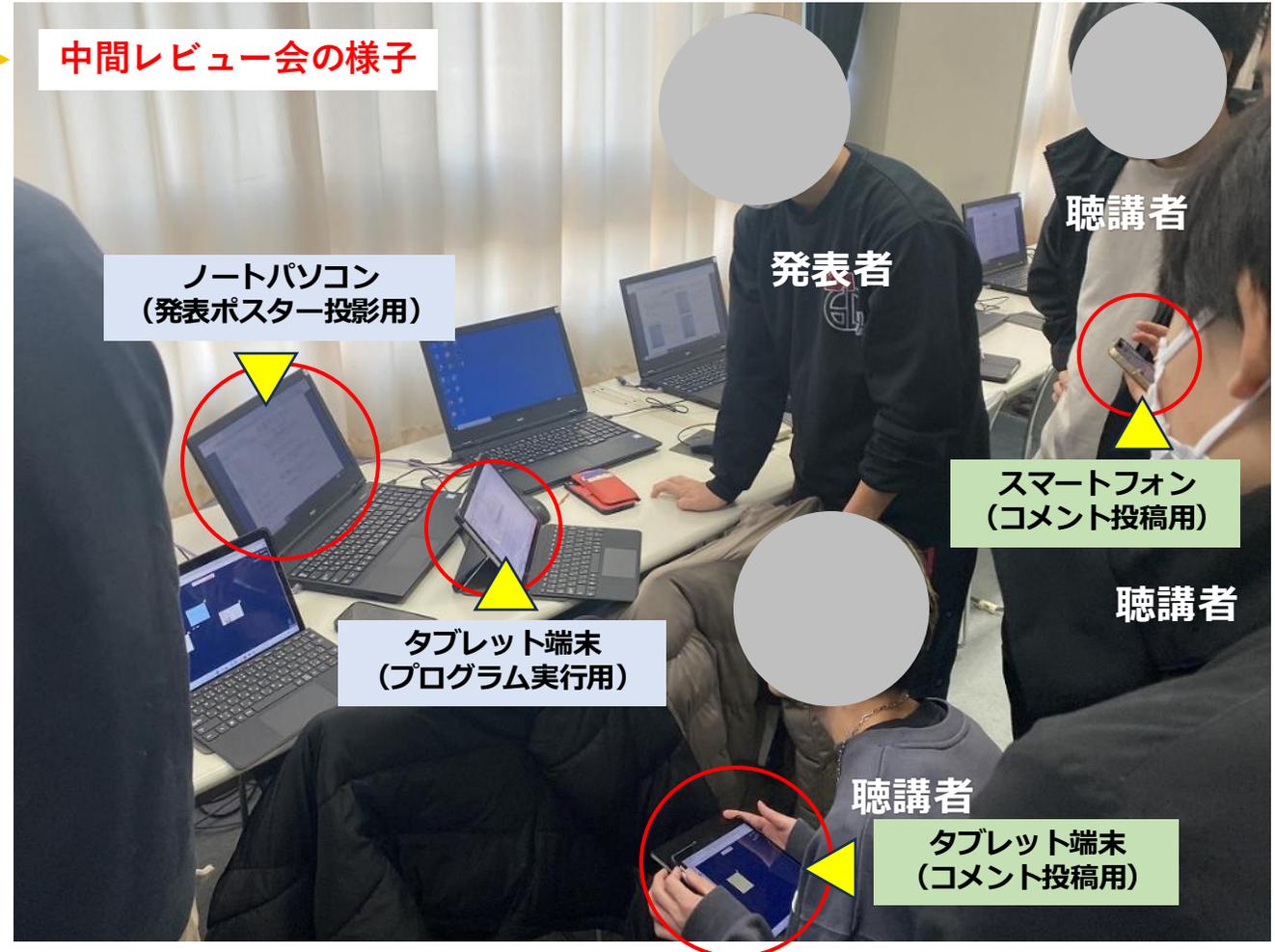


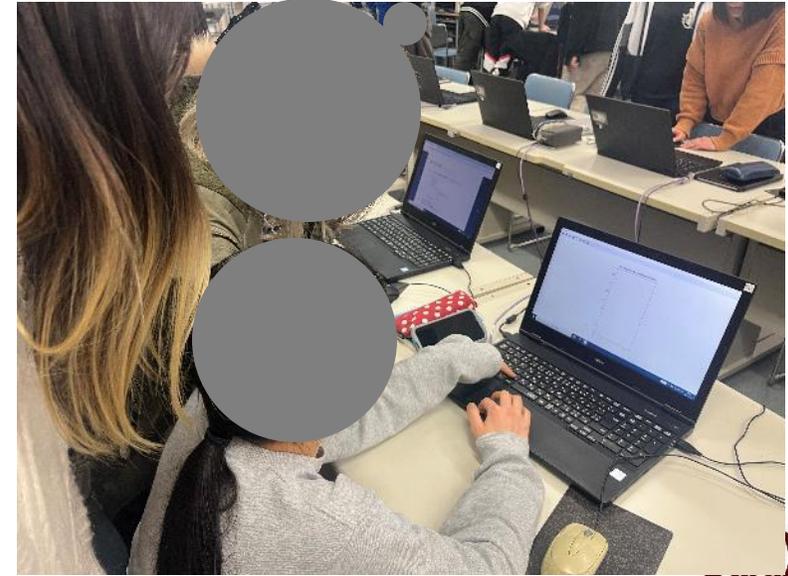
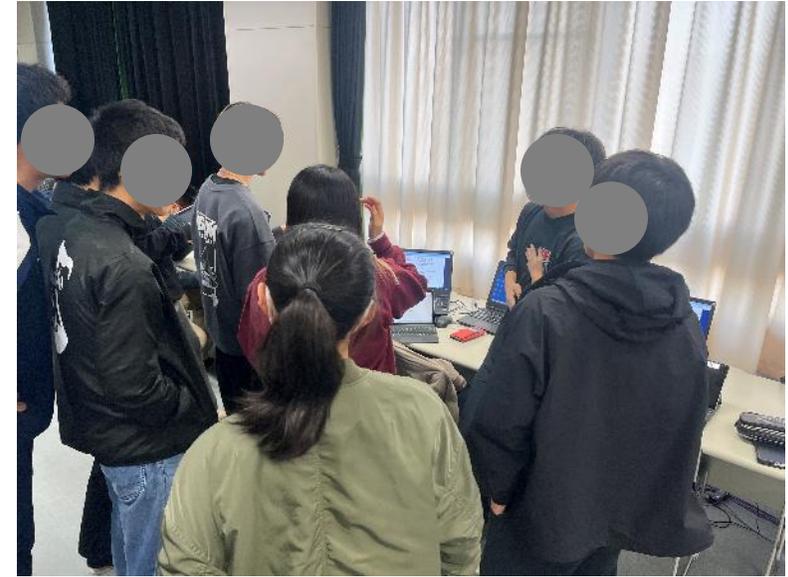
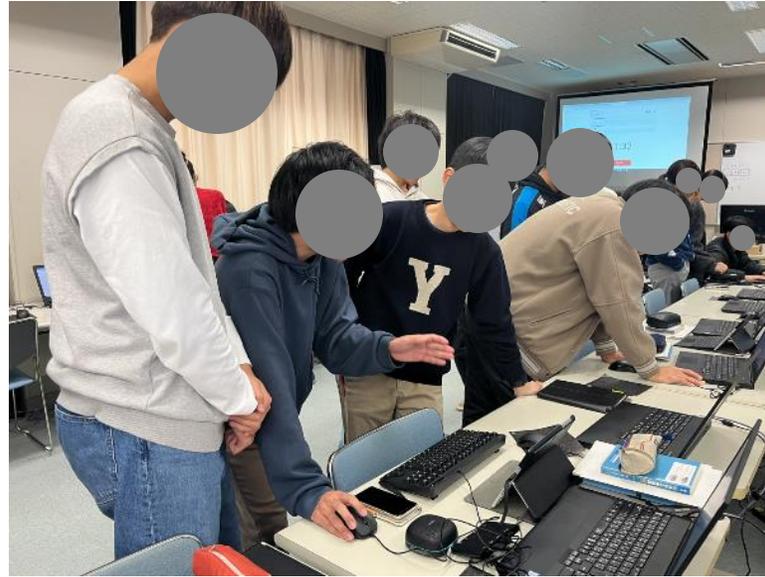

Python-Day3 条件分岐



# 中間レビュー会の様子

授業回	授業内容
1	メッセージ
2	変数
3	条件分岐
4	前半 繰り返し（for文）
5	8回 繰り返し（while文）
6	配列／リスト（基礎）
7	配列／リスト（応用）
8	関数
9	後半 プログラム制作①
10	プログラム制作②
11	プログラム制作③
12	6回 中間レビュー会
13	プログラムの評価・改善
14	プログラム発表会





ポスター提出【最終版】

名前検索 回答を隠す 回答共有中 一括返却

画面記憶

◎ 1月28日(火) 17:00 比較

ポスター提出のサムネイル一覧:

- ハンドクリーム診断 (1月21日(火) 13:40)
- ロコミスクレイピング (1月20日(月) 23:27)
- ブロックスで遊ぼう (1月21日(火) 11:37)
- 今日はどんな練習をしようかな? (1月21日(火) 13:06)
- 旭丘シミュレーションゲーム (1月21日(火) 13:53)
- ギターの音を覚えよう (1月14日(火) 13:48)
- 服装提案プログラム (1月20日(月) 21:48)
- テンテロで遊ぼう!! (1月21日(火) 0:18)
- カレンダーと予定を表示する (1月21日(火) 13:01)
- ToDoリスト (1月21日(火) 12:47)
- To doリスト (1月20日(月) 20:06)
- 今日のラッキーカラー (1月20日(月) 21:10)
- 今日のラッキーカラー (1月21日(火) 13:00)
- 星座占い (1月21日(火) 8:25)
- 素因数分解 (1月14日(火) 13:47)
- モンキーハンティング (1月14日(火) 13:46)
- ToDoリストで自分を律しよう (1月21日(火) 13:45)
- ポーカー (1月21日(火) 12:57)
- 概要 (1月21日(火) 13:41)
- エアホッケー (1月21日(火) 13:40)
- 動画活用管理のプログラミング (1月20日(月) 22:30)
- パーソナルカラー診断 (1月14日(火) 13:52)
- 2025年恋愛運 (1月20日(月) 16:38)
- 須江を探せ!! (1月20日(月) 21:50)
- イデオロギー診断 (1月21日(火) 10:32)
- エアコンのマイナット (1月21日(火) 23:44)
- 愛衣打 (1月21日(火) 13:39)
- 大喜利の方々 (1月21日(火) 13:38)
- 希望通りにプログラム (1月21日(火) 13:37)
- Pythonでゼロ (1月21日(火) 13:36)
- セルもぐらたき (1月21日(火) 13:35)
- 皆さんの認知機能を記憶力をもとにチェックします (1月21日(火) 13:34)
- ガチャシミュレーター (1月21日(火) 13:33)



# 早く家に帰りたい！

家に帰るためのバスが3駅から出ているので  
そのうちのどのバスが待ち時間が少ないのか  
最適なバスを表示してくれるプログラム  
きっと需要は私にしかない



←こんな感じ



```
HTML
<html lang="ja">
  <body onload="JikokuChange()">
    <select id="Imaike" name="Imaike" onchange="JikokuChange()"></select>
    <div id="Narukokita"></div>
    <div id="Aoiyama"></div>
    <div id="Kamisawa"></div>
    <div id="Saisoku"></div>
  </body>
</html>

JavaScript
const Imaike=["05:31","05:42","05:53",
"06:03","06:13","06:23","06:31","06:39","06:40","06:53",
"07:00","07:06","07:12","07:18","07:25","07:28","07:33","07:38","07:45","07:48","07:55","07:58",
"08:05","08:07","08:11","08:15","08:19","08:23","08:27","08:31","08:35","08:39","08:43",
"08:47","08:51","08:55",
"09:00","09:05","09:09","09:14","09:19","09:24","09:30","09:37","09:45","09:52",
"10:00","10:07","10:15","10:25","10:35","10:45","10:55",
"11:00","11:15","11:25","11:35","11:45","11:55",
"12:00","12:15","12:25","12:35","12:45","12:55",
"13:05","13:15","13:25","13:35","13:45","13:55",
"14:05","14:15","14:25","14:35","14:45","14:55",
"15:05","15:15","15:25","15:35","15:40","15:40","15:55",
"16:05","16:10","16:18","16:25","16:33","16:40","16:48","16:55",
"17:05","17:10","17:18","17:24","17:30","17:36","17:42","17:48","17:54",
"18:00","18:06","18:12","18:18","18:24","18:30","18:36","18:42","18:48","18:54",
"19:00","19:06","19:12","19:20","19:27","19:35","19:42","19:50","19:57",
"20:05","20:12","20:20","20:27","20:35","20:42","20:50","20:57",
"21:06","21:15","21:25","21:35","21:45","21:55",
"22:05","22:15","22:27","22:39","22:51"];
const narukokita=["06:15","07:02","07:14","07:32","07:55","08:00","08:20","08:32","08:50",
"09:02","09:42","09:55","10:22","10:55","11:02","11:42","11:55","12:22","12:55","13:02","13:42",
"13:55","14:22","14:55","15:02","15:42","15:55","16:12","16:42","16:55","17:11","17:40","17:47",
"18:15","18:22","18:45","18:59","19:33","19:49","20:20","20:40","21:17","22:05"];
const aoiyama=["06:24","06:40","07:00","07:15","07:38","08:02","08:17","08:55","08:55",
"09:17","09:44","10:27","11:12","12:07","13:07","14:07","15:07","16:07","17:09","18:10",
"18:49","19:29","20:04","20:49","21:59","22:59"];
const kamisawa=["08:29","09:29","10:29","11:29","12:29","13:29","14:29","15:29","16:29"];
// 時刻表の取得
const select=document.querySelector('select[name="Imaike"]');
Imaike.forEach(ImaieTime=>{
  var option=document.createElement('option');
  option.text=ImaieTime;
  option.value=ImaieTime;
  select.add(option);
});
// 現在の時刻から一番近い時刻を選択する
var element=document.getElementById('Imaike');
var options=element.options;
var d=new Date();
for(var i=0;i<options.length;i++){
  var TrainTime=options[i].value;
  if(d.getHours()<TrainTime.slice(0,2)||d.getHours()>TrainTime.slice(0,2)&&
  d.getMinutes()>TrainTime.slice(-2))){
    options[i].selected=true;
    break;
  }
}
function JikokuChange()
{
  const Chikatetsu=document.getElementById("Imaike").value;
  var ChikaDate=new Date();
  var DateNow=new Date();
  ChikaDate=new Date(DateNow.getFullYear(),DateNow.getMonth(),DateNow.getDate(),
  Chikatetsu.slice(0,2),Chikatetsu.slice(-2),0);
  //鳴子北の時刻表と比較する
  ChikaDate.setMinutes(ChikaDate.getMinutes()+24);
  var narukobus="56:00";
  for(var i=0;i<narukokita.length;i++){
    if(ChikaDate.getHours()<narukokita[i].slice(0,2)||
    (ChikaDate.getHours()==narukokita[i].slice(0,2)&&
    ChikaDate.getMinutes()<narukokita[i].slice(-2)))
    {
      narukobus=narukokita[i];
      break;
    }
  }
  var divNaruko=document.getElementById("Narukokita");
  if(narukobus!="56:00"){
    divNaruko.innerHTML="鳴子北"+narukobus;
  }
  else{divNaruko.innerHTML="バスはありません";}
  //相生山の時刻表と比較する
  ChikaDate.setMinutes(ChikaDate.getMinutes()+2);
  var aoiibus="56:00";
  for(var i=0;i<aoiyama.length;i++){
    if(ChikaDate.getHours()<aoiyama[i].slice(0,2)||
    (ChikaDate.getHours()==aoiyama[i].slice(0,2)&&
    ChikaDate.getMinutes()<aoiyama[i].slice(-2)))
    {
      aoiibus=aoiyama[i];
      break;
    }
  }
  var divAioi=document.getElementById("Aoiyama");
  if(aoiibus!="56:00"){
    divAioi.innerHTML="相生山"+aoiibus;
  }
  else{divAioi.innerHTML="バスはありません";}
  //神沢の時刻表と比較する
  ChikaDate.setMinutes(ChikaDate.getMinutes()+2);
  var kamibus="56:00";
  for(var i=0;i<kamisawa.length;i++){
    if(ChikaDate.getHours()<kamisawa[i].slice(0,2)||
    (ChikaDate.getHours()==kamisawa[i].slice(0,2)&&
    ChikaDate.getMinutes()<kamisawa[i].slice(-2)))
    {
      kamibus=kamisawa[i];
      break;
    }
  }
  var divKami=document.getElementById("Kamisawa");
  if(kamibus!="56:00"){
    divKami.innerHTML="神沢"+kamibus;
  }
  else{divKami.innerHTML="バスはありません";}
  //一番早いバスを表示する
  var divSaisoku=document.getElementById("Saisoku");
  if(narukobus<aoiibus&&narukobus<kamibus){
    divSaisoku.innerHTML="鳴子北"+narukobus+"が最速!";
  }else if(aoiibus<narukobus&&aoiibus<kamibus){
    divSaisoku.innerHTML="相生山"+aoiibus+"が最速!";
  }else if(kamibus<narukobus&&kamibus<aoiibus){
    divSaisoku.innerHTML="神沢"+kamibus+"が最速!";
  }else{
    divSaisoku.innerHTML="歩いて帰れ!";
  }
}
```



# 急行

苦労点(聞いてほしい)：音源が見つからず、拳句の果てに名古屋駅の放送音源を高蔵寺・神宮前・金山・西春・須ヶ口・知立駅放送から錬成しました。

1  
号車

## まもなく シン・鉄道発車標総合表示システム(ME-R7-3)

このボタンでモードの切り替えができます！！

仕組みの概要(名鉄→JRにするとき)

- ①モニターの中にある名鉄用の図形をすべて選択し消去する。
- ②右のコードで他のシートに張り付けてあるJR用の図形をすべて選択し、指定の位置に貼り付ける。

JR民の方、長らくお待たせしました  
祝 JRモードサービス提供開始！！

```
Sub ShapeCopy6()
    Dim shape As shape
    Dim area1 As Range
    Dim area2 As Range
    Dim area3 As Range

    Worksheets("メイン").Activate
    Set area1 = Range("B7:V20")

    For Each shape In ActiveSheet.Shapes
        If Not Intersect(shape.TopLeftCell, area1) Is Nothing Then
            shape.Delete
        End If
    Next shape
(略)
Worksheets("メイン").Select
For Each myShape3 In Worksheets("コピー先 (2)").Shapes
    myShape3.Copy
    Worksheets("メイン").Paste
    Selection.Top = myShape3.Top
    Selection.Left = myShape3.Left
Next

Worksheets("メイン").Activate
Dim myShape4 As Variant
Worksheets("メイン").Select
For Each myShape4 In Worksheets("コピー先 (4)").Shapes
    myShape4.Copy
    Worksheets("メイン").Paste
    Selection.Top = myShape4.Top
    Selection.Left = myShape4.Left
Next
End Sub
```



団体名	文常	合計	¥28,500	フィルターリセット
		予算残額	(¥7,470)	
年月日	2024/12/1 から	2024/12/31 まで		日付フィルター
購入項目	ab	項目フィルター		
申請回	第二回	申請フィルター		印刷

文常申請第二回 青紙番号	入力日	項目	単価	個数	金額	
1	100	2024/12/3 a	¥300	1	¥300	第一回
2	100	2024/12/5 a	¥300	1	¥300	第一回
3	100	2024/12/5 a	¥300	1	¥300	第一回
4	100	2025/1/16 a	¥3,000	1	¥3,000	第一回
5	100	2025/1/16 b	¥2,000	1	¥2,000	第一回
6	100	2025/1/16 e	¥700	3	¥2,100	第一回
7	99	2025/1/16 ab	¥500	5	¥2,500	第二回
8	99	2025/1/16 ad	¥3,000	1	¥3,000	第二回
9	99	2025/1/16 ak	¥5,000	3	¥15,000	第二回
10						
--						

支給	青紙番号	仮入力カウンター	総仮入力回数
第二回	99	現在0回	現在9回

団体名	物品名	単価	個数	一般

消費税	団体名	金額	消費税	仮入力巻き戻し

振込手数料	団体名	銀行	30000円以上か否か	振込手数料	クリア

派遣費	団体名	項目	金額	件数	派遣費

会計システム 入力を簡素化し一覧性を向上させた

```

End If
GoTo Theend
Error:
MsgBox "フィルター項目を入力してください"
Theend:

End Sub
Sub 申請フィルター()

If Sheets("検索システム").Range("C7").Value = 0 Then
GoTo Error
Else
Dim a
a = Sheets("検索システム").Range("C7")
Dim y
'Debug.Print Application.WorksheetFunction.CountIf(Sheets("保存先").Range("D:D"), "*")
For y = 10 To Application.WorksheetFunction.CountIf(Sheets("保存先").Range("D:D"), "*") + 9

```

```

If Sheets("入力").Range("B1").Value = 0 Then
MsgBox "仮入力を行ってください"
Else
i = Sheets("入力").Range("C1")
Dim c
For c = N - M + 1 To N + 1 + 1
Sheets("保存先").Cells(c, 1).Value = Sheets("メモリ").Cells(c - 1, 1)
Sheets("保存先").Cells(c, 2).Value = 1 + 1
Sheets("保存先").Cells(c, 4).Value = Sheets("メモリ").Cells(c - 1, 2)
Sheets("保存先").Cells(c, 5).Value = Sheets("メモリ").Cells(c - 1, 3)
Sheets("保存先").Cells(c, 6).Value = Sheets("メモリ").Cells(c - 1, 7)
Sheets("保存先").Cells(c, 7).Value = Sheets("メモリ").Cells(c - 1, 4)
Sheets("保存先").Cells(c, 8).Value = Sheets("メモリ").Cells(c - 1, 5)
Sheets("保存先").Cells(c, 9).Value = Sheets("保存先").Cells(c, 7) * Sheets("保存先").Cells(c, 8)
Sheets("保存先").Cells(c, 10).Value = Date
Sheets("保存先").Cells(c, 10).Font.Color = RGB(255, 255, 255)
Sheets("保存先").Cells(c, 11).Value = Sheets("入力").Range("A3").Value
Sheets("保存先").Cells(c, 11).Font.Color = RGB(255, 255, 255)
Sheets("保存先").Cells(c, 3).Value = Application.WorksheetFunction.CountIf(Range(Sheets("保存先").Cells(2, 4), Sheets("保存先").Cells(c, 4)), Sheets("保存先").Cells(c, 4)) & Sheets("保存先").Cells(c, 4)
Sheets("保存先").Cells(c, 3).Font.Color = RGB(255, 255, 255)
Application.StatusBar = c & "/" & M + 1 + 1
Application.StatusBar = False
Next c
Sheets("保存先").Range(Sheets("保存先").Cells(N + 1 + 2, 1), Sheets("保存先").Cells(N + 1 + 2, 9)).Interior.Color = RGB(169, 169, 255)
Sheets("保存先").Cells(N + 1 + 2, 1).Value = Sheets("入力").Range("A3").Value
If Sheets("保存先").Cells(N + 1 + 1, 1).Value = "" Then
Sheets("保存先").Cells(N + 1 + 2, 2).Value = Sheets("保存先").Cells(N + 1 + 1, 4)
Else
Sheets("保存先").Cells(N + 1 + 2, 2).Value = Sheets("保存先").Cells(N + 1 + 1, 1)
End If
Sheets("保存先").Cells(N + 1 + 2, 3).Value = "入力日"
Sheets("保存先").Cells(N + 1 + 2, 4).Value = Date
Sheets("保存先").Cells(N + 1 + 2, 5).Value = "合計金額"
Sheets("保存先").Cells(N + 1 + 2, 6).Value = Application.WorksheetFunction.Sum(Sheets("保存先").Range(Sheets("保存先").Cells(N - M + 1 + 2, 9), Sheets("保存先").Cells(N + 1 + 1, 9)))
Sheets("保存先").Cells(N + 1 + 2, 7).Value = "予算残額"
Sheets("保存先").Cells(N + 1 + 2, 8).Value = Application.WorksheetFunction.VLookup(Sheets("保存先").Cells(N + 1 + 2, 2), Sheets("予算表").Range("A1:B300"), 2, False) - Application.WorksheetFunction.
If Sheets("保存先").Cells(N + 1 + 2, 8).Value < 0 Then
Sheets("保存先").Cells(N + 1 + 2, 8).Interior.Color = RGB(255, 0, 0)
Else

```



# パスカルの三角形の合同式での塗り分けによるフラクタル図形の作成(変更版)

中間発表のときから色を変更

mod 5の場合

```

size = 100
p = 5 ' mod 5
' シートをクリア
Dim ws As Worksheet
Set ws = ThisWorkbook.Sheets(4)
ws.Cells.Clear
' 配列を初期化
ReDim row(1 To size)
row(1) = 1 ' 最初の行の最初の値を1に設定
' 初期の開始列を設定（中央揃え）
startCol = size

```

```

' パスカルの三角形
For i = 1 To size
    Dim prevRow() As Integer
    prevRow = row ' 現在の行を前の行として保持
    ' 現在の行を計算
    For j = 1 To i
        If j = 1 Or j = i Then
            row(j) = 1 ' 両端は1
        Else
            row(j) = (prevRow(j - 1) + prevRow(j)) Mod p ' 上2つのセルを加算
        End If
    End For

```

```

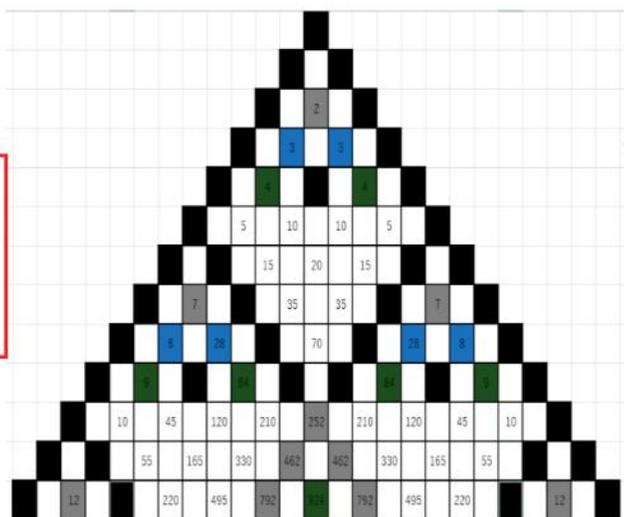
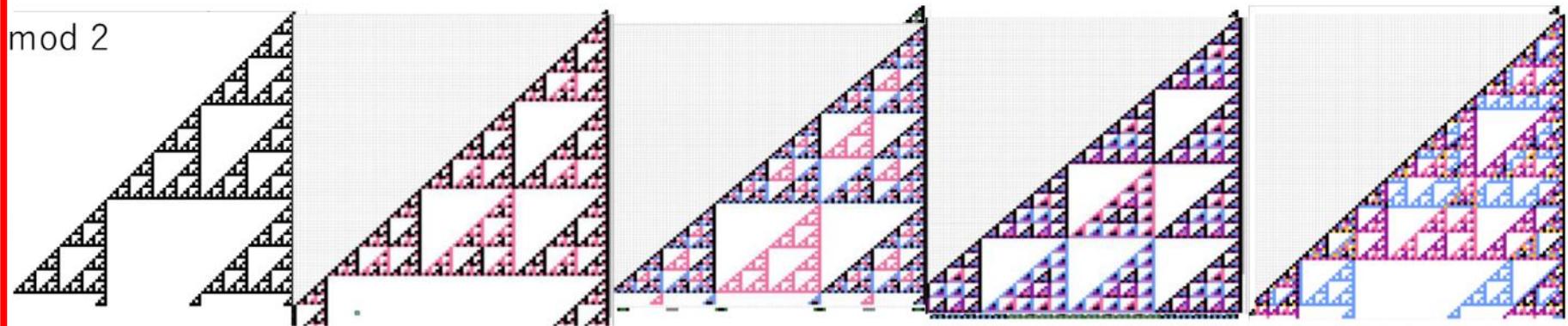
' modで塗り分け
Select Case row(j)
Case 1
    ws.Cells(i, startCol + j - 1).Interior.Color = RGB(0, 0, 0) ' 黒
Case 2
    ws.Cells(i, startCol + j - 1).Interior.Color = RGB(128, 128, 128) ' 灰
Case 3
    ws.Cells(i, startCol + j - 1).Interior.Color = RGB(25, 107, 36)
Case 4
    ws.Cells(i, startCol + j - 1).Interior.Color = RGB(77, 147, 217)
Case Else
    ws.Cells(i, startCol + j - 1).Interior.Color = RGB(255, 255, 255) ' 白
End Select

```

```

' modで塗り分け
Select Case row(j)
Case 1
    ws.Cells(i, startCol + j - 1).Interior.Color = RGB(0, 0, 0) ' 黒
Case 2
    ws.Cells(i, startCol + j - 1).Interior.Color = RGB(251, 107, 162)
Case 3
    ws.Cells(i, startCol + j - 1).Interior.Color = RGB(102, 153, 255)
Case 4
    ws.Cells(i, startCol + j - 1).Interior.Color = RGB(153, 0, 153) ' 紫
Case Else
    ws.Cells(i, startCol + j - 1).Interior.Color = RGB(255, 255, 255)
End Select
Next j
Next i
End Sub

```



【合同式】

$$7 \equiv 4 \pmod{3}$$

⇨ 7と4は3で割ったときの余りが等しい

【手順】

- ①パスカルの三角形を描く
- ②xで割ったとき(mod x)の余りごとに色分け

←手打ちで描いたもの



## 発表 最終版 「素数の分布の視覚化」

### <概要>

エラトステネスのふるいの考え方をを用いて効率的に素数の配列を作り、その要素と要素が何番目かをすべて表示する。座標平面は縦軸は素数の大きさ、横軸は何番目の素数かを表す。

### <プログラムの要約>

hanni=〇〇まで調べるとき、  
 (これを自由に変えてグラフの表示範囲を変えられる)  
 配列primesを定義し、最初の素数2を含めておく。  
 3から調べるhanniまでの素数を配列primesに含めていく。  
 hantei=1を代入して、 $\sqrt{i}$ までのprimesの要素で割っていく。  
 3から調べるhanniまでの素数を配列primesに含めていく。  
 素数でないと確定すればその時点でbreakをし、次のiを調べる。  
 以上の操作により、配列を用意したら後はx番目の素数をyとして、プロット  
 <グラフを修飾・表示するプログラム>

### <工夫>

グーグルコロボを使ってpythonの機能を拡張して座標平面を使えるようにし、プログラムを組んだ。  
 素数判定のプログラムでは $\sqrt{n}$ までの自然数で順に割るようにし、無駄な計算を極限まで省くことで処理を軽くした。  
 エラトステネスのふるいの手法を繰り返して順次利用していき、配列に素数を追加する。中間発表のプログラムに比べて大きく効率が向上している。

### <改善点>

hanniを1000万にすると100万に比べられないほどの時間がかかる。  
 (1分54秒)処理時間が早まったとはいえ、大きな数になるとその影響はわからないほど小さくなってしまう。Hanniの値によってふるいの大きさを最適化するプログラムが最も早いと考えられる。(セグメントふるい)

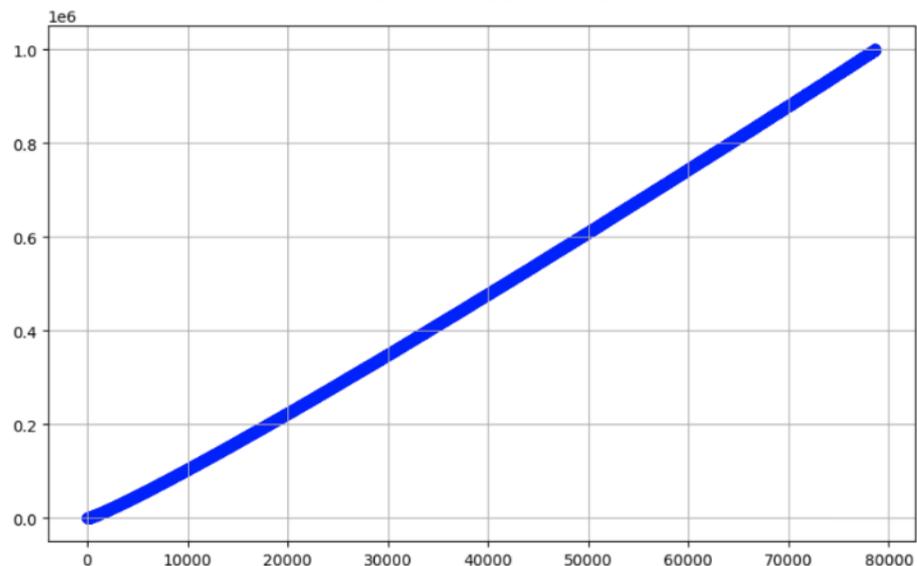
### <プログラム> (グーグルコロボのpython)

```
import matplotlib.pyplot as plt
hanni=10**6
primes=[2]
for i in range(3,hanni+1):
    hantei=1
    for j in range(1,len(primes)):
        if i%(primes[j-1])==0:
            hantei=0
            break
    if primes[j-1]>=i**(1/2):
        break
~~~~~
if hantei==1:
    primes.append(i)
x=[]
y=[]
for i in range(1,len(primes)+1):
    y.append(primes[i-1])
    x.append(i)plt.figure(figsize=(10,6))
plt.scatter(x,y,color='blue',label='prime')
plt.grid(True)
plt.show
```

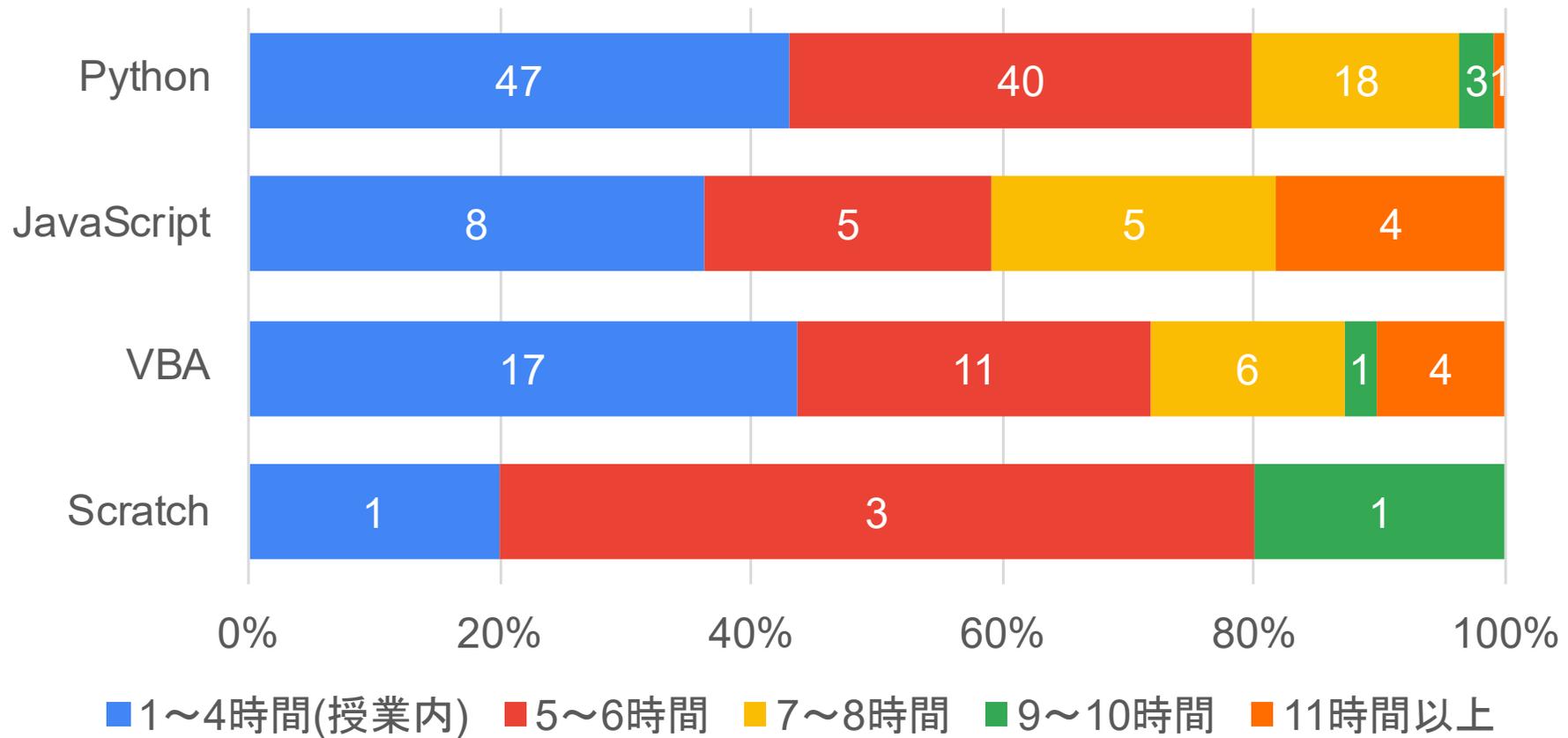
### <実行結果>

\* 100万までの素数で調べると、改良前(中間発表参照)が13秒かかったのに対し、改良後(上に示したプログラム)は7秒で終了し、中間発表での改善点を克服できた。

以下はhanni=10\*\*6=100万での出力である。

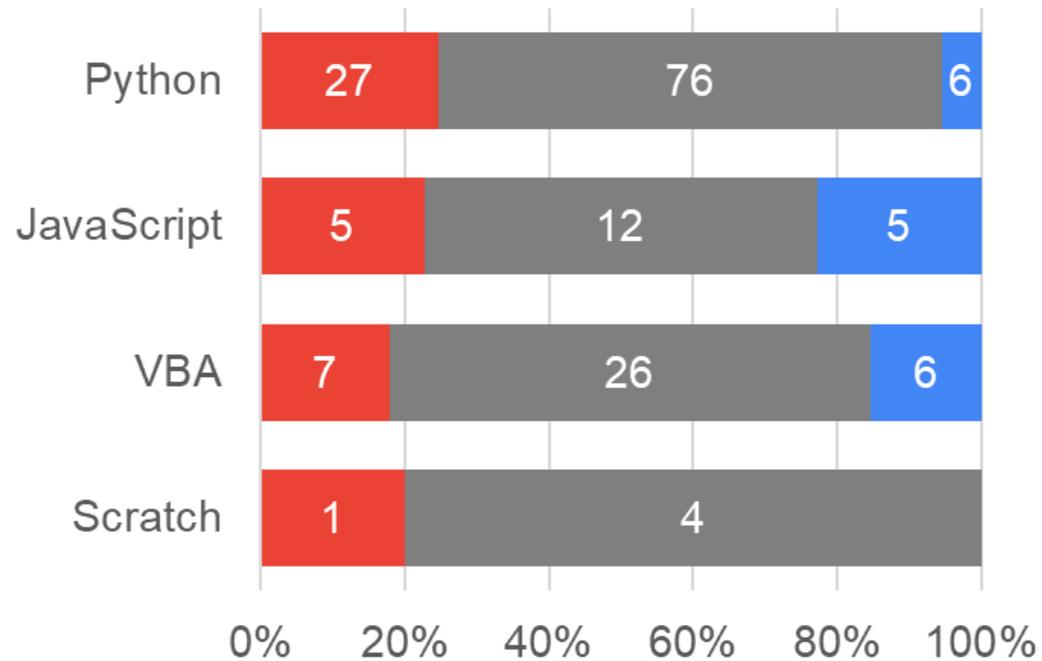


## プログラム制作にかけた時間

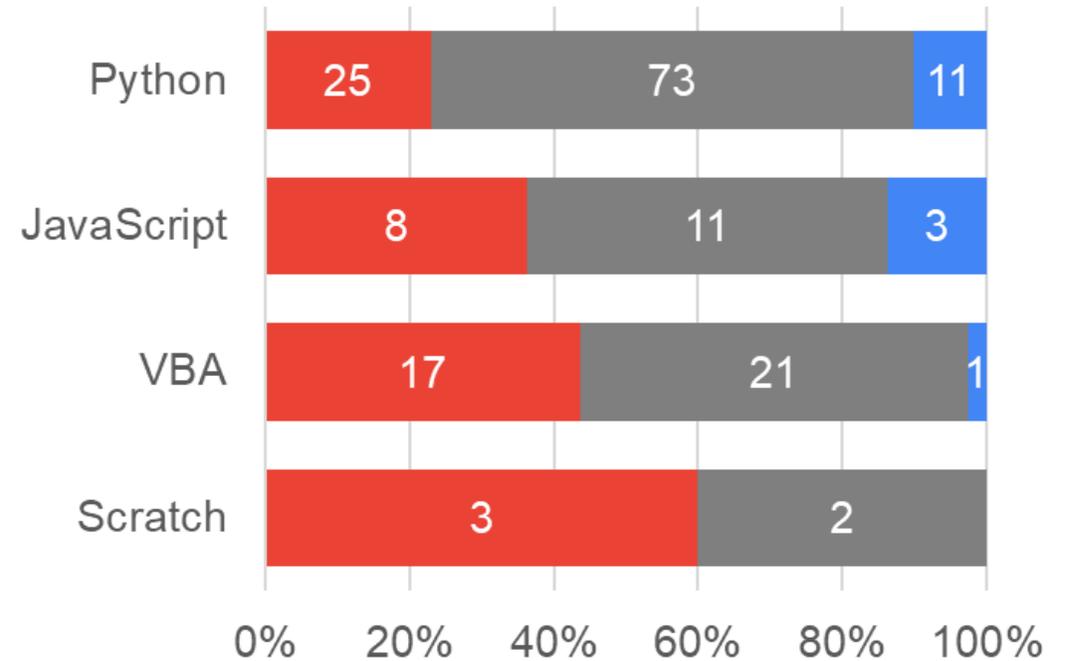


# プログラミングの授業回数について

## 【前半】プログラミングの基礎（8回）



## 【後半】プログラム制作と発表会（6回）



■ もっと多い方がいい ■ ちょうどいい ■ もっと少ない方がいい

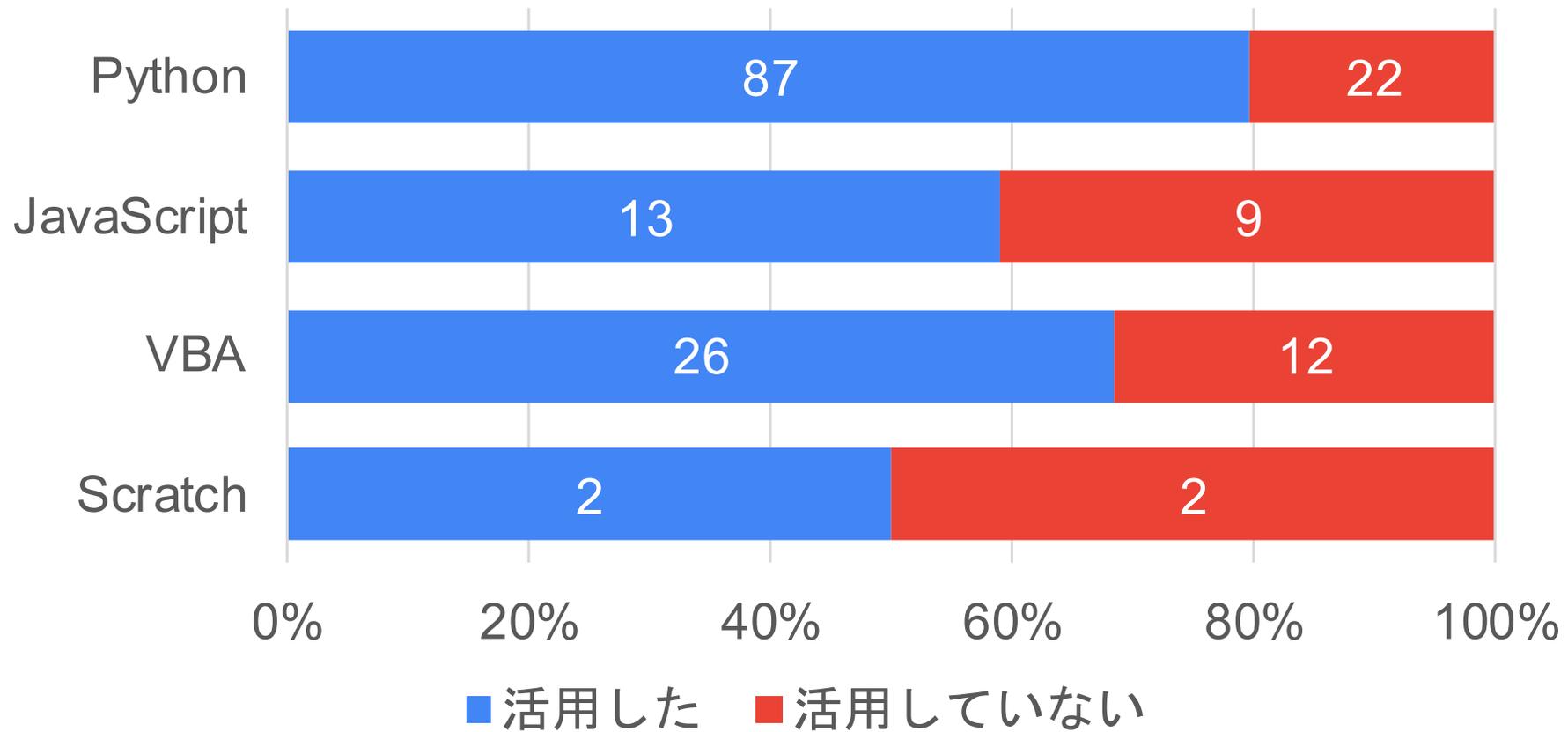


## プログラミングにおける生成AIの利用

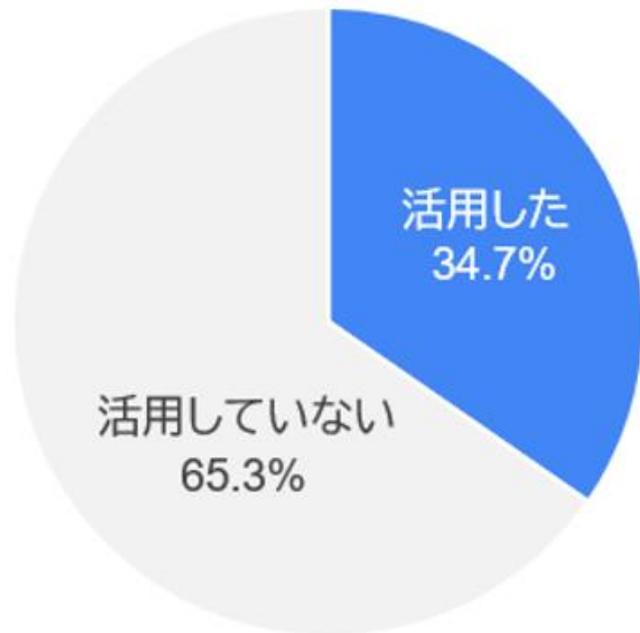
- 【後半】 のはじめに生成AIの活用について説明
  - 🎯 アイデアの提案
  - 🎯 プログラムの生成
  - 🎯 プログラムの修正（デバッグ）
- 生成AIの使用は任意
- 生徒全員，保護者からの同意書を提出済み



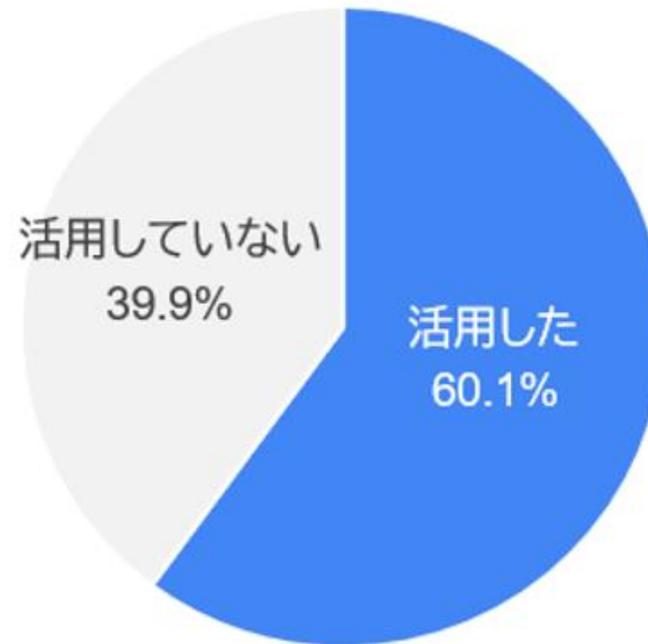
## プログラム制作で生成AIを活用したか



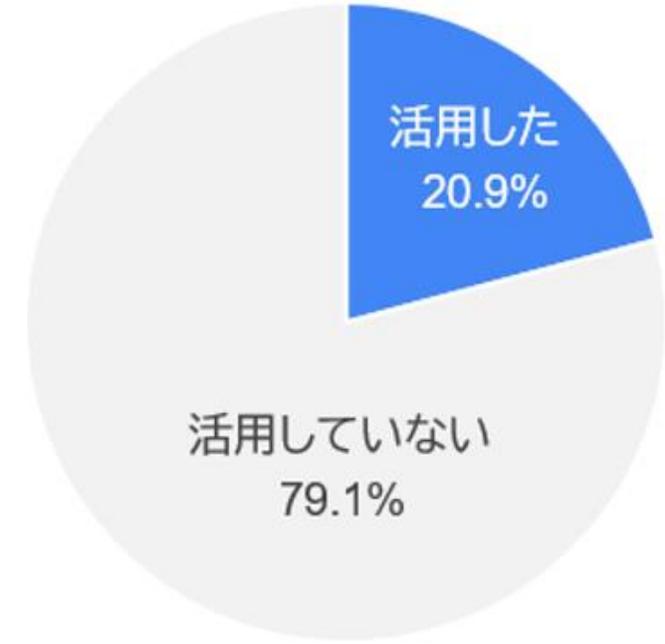
## 生成AI活用の活用場面（複数回答可）



(b) ①アイデアの提案



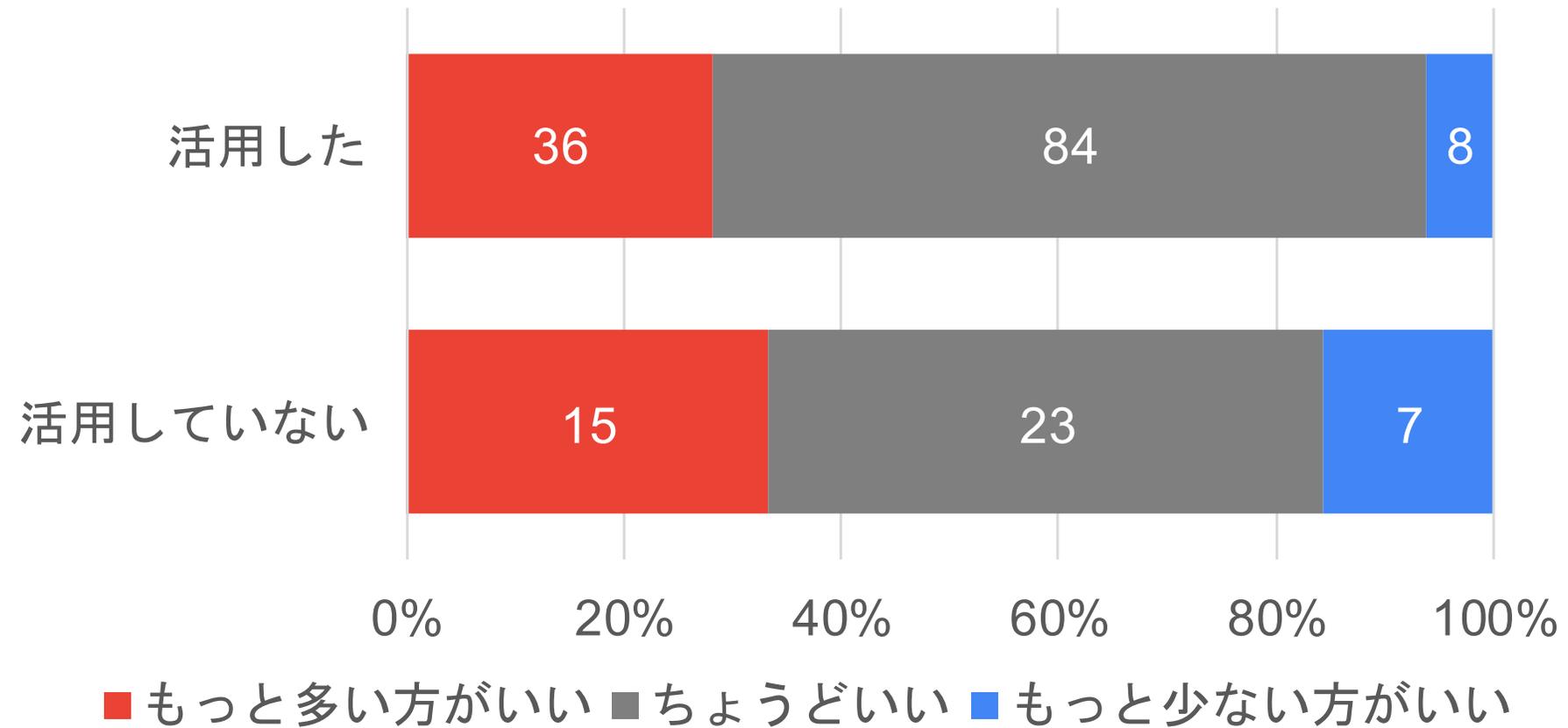
(c) ②プログラムの生成



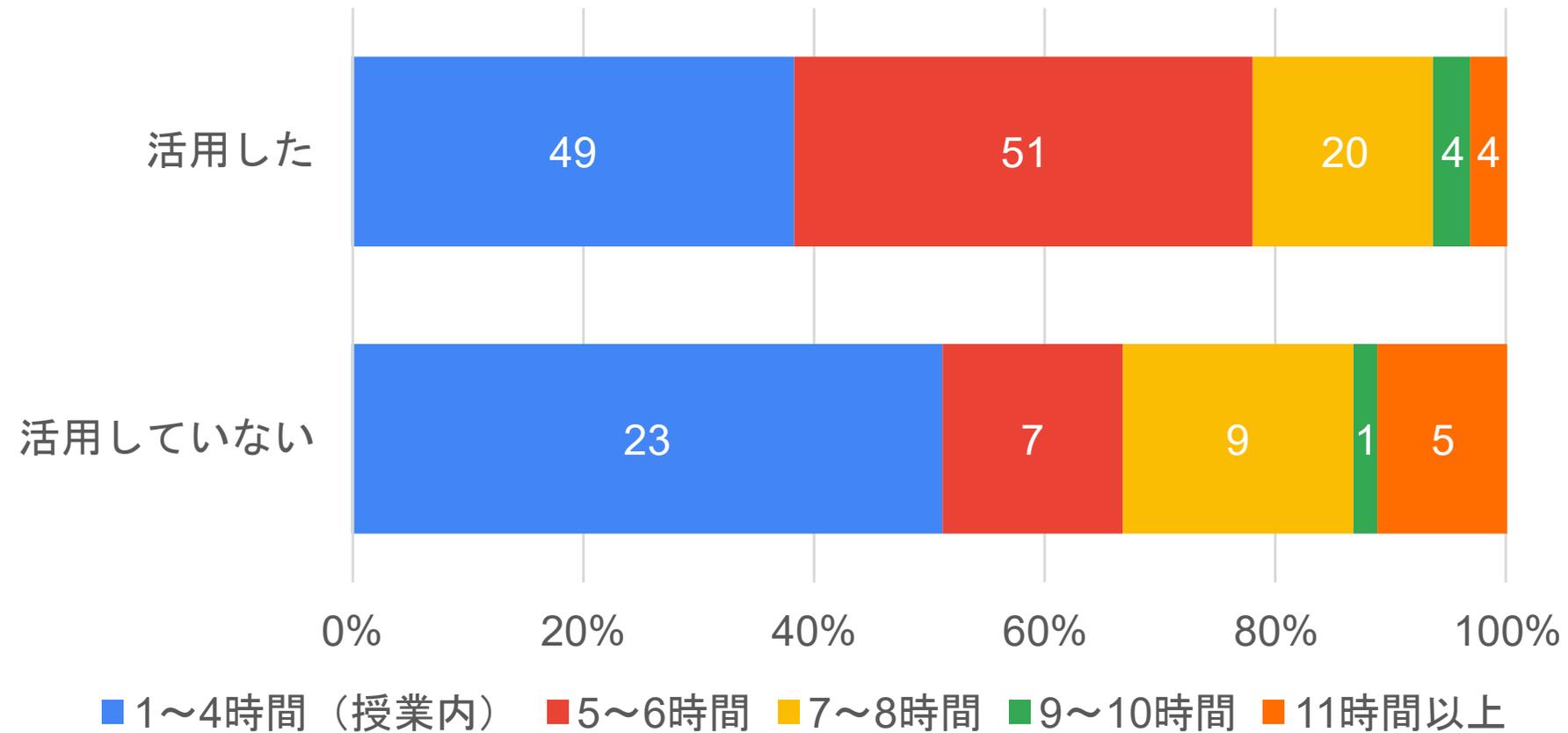
(d) ③プログラムの修正



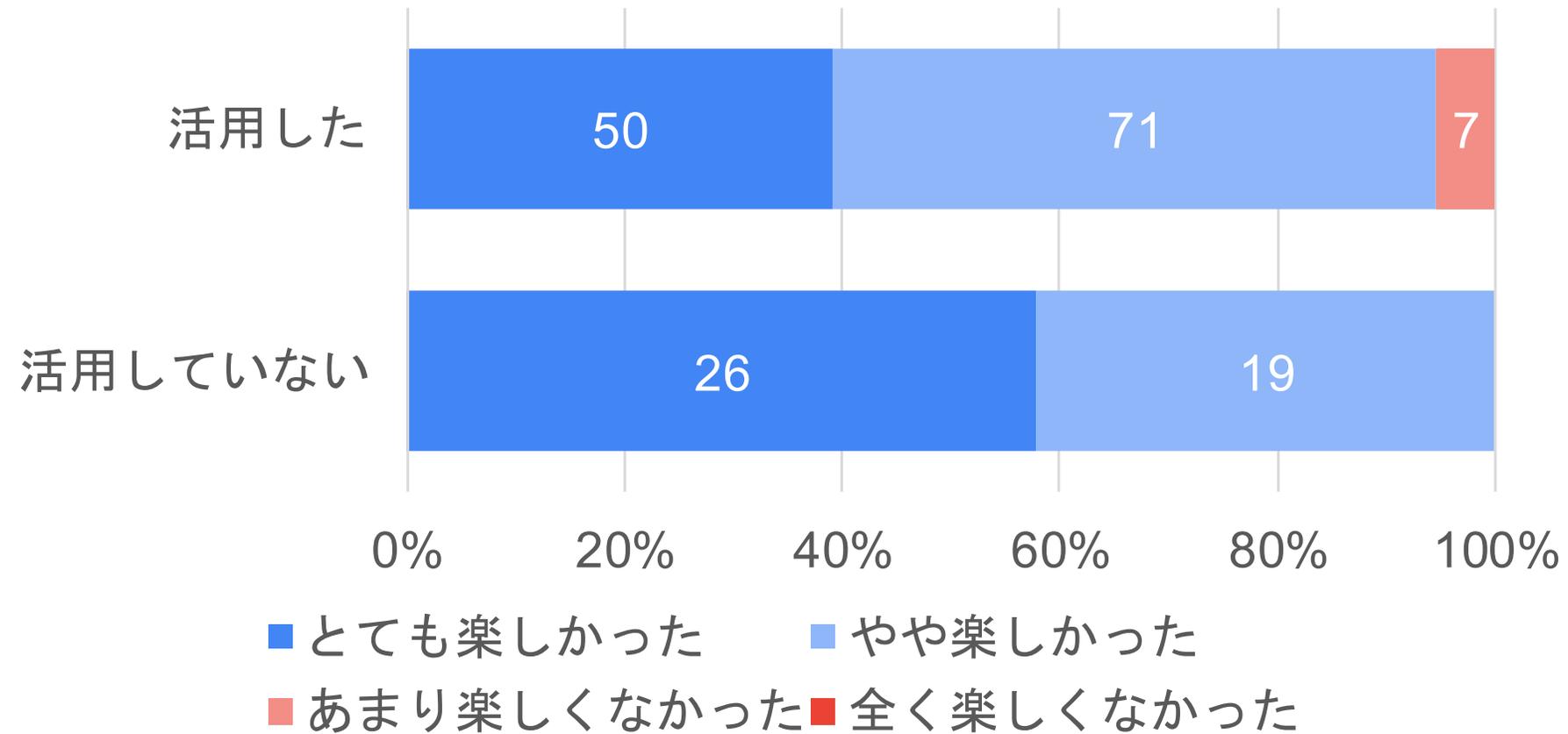
## 生成AIの活用×授業回数



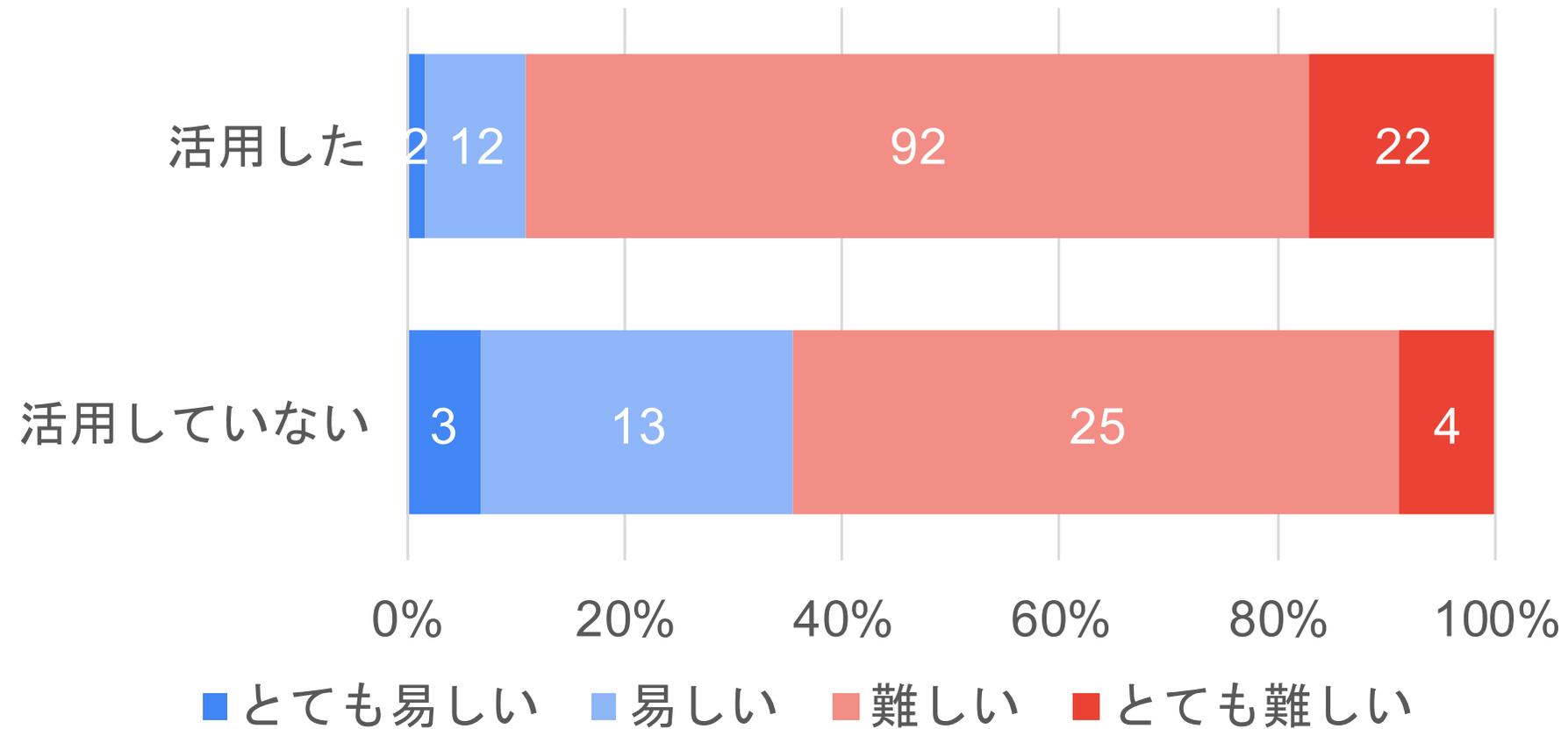
## 生成AIの活用×プログラム制作の時間



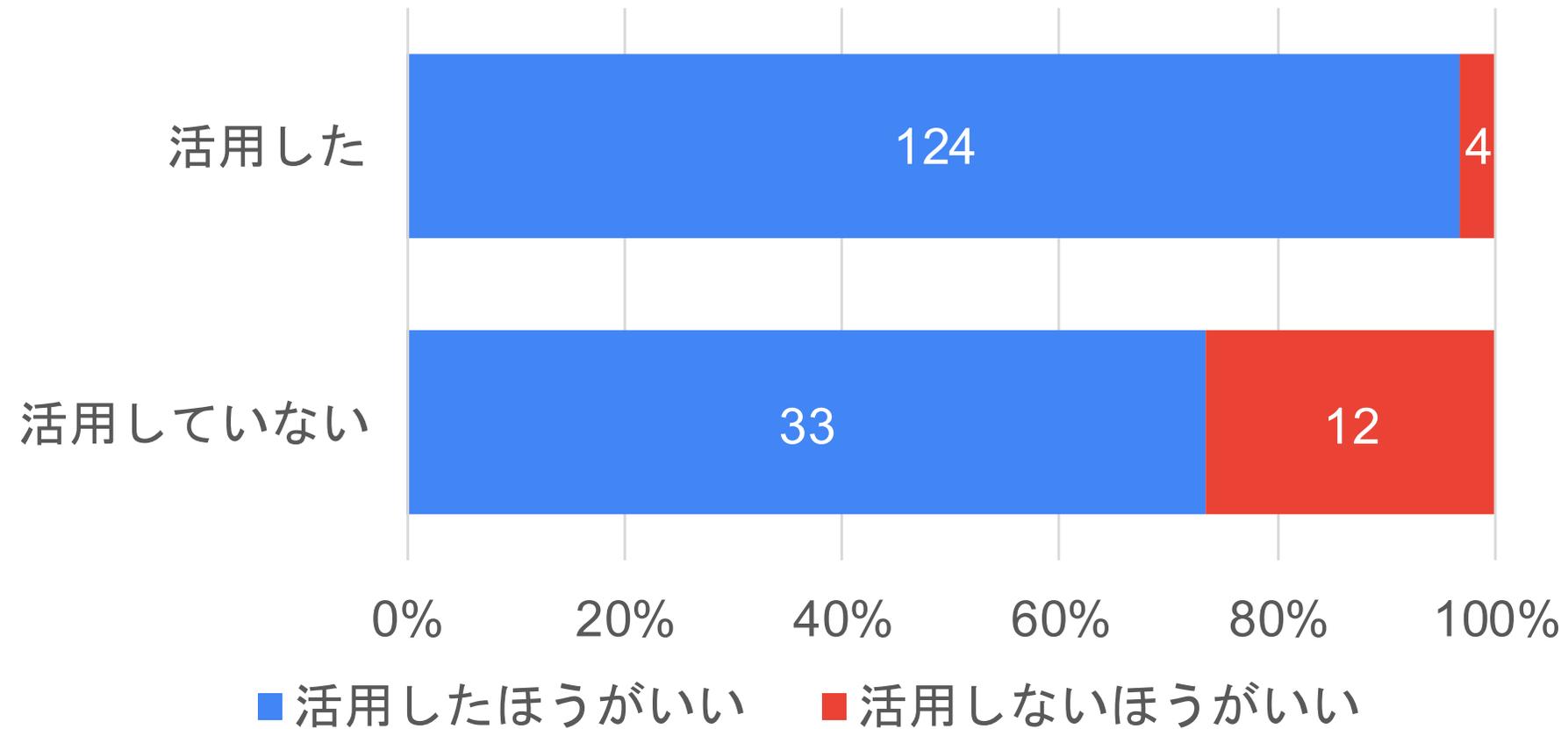
## 生成AIの活用×プログラミングの楽しさ



## 生成AIの活用×プログラミングの難しさ



## 生成AIの活用 × 生成AIの活用の是非



## 生成AIに関する自由記述

- プログラミングの配列など独特の考え方はかなり練習しないと自分の作品作りには使いにくいと感じました。その一方で、**チャットGPTなどを使って自分のアイデアを大まかにプログラムしてもらい、細かな修正を自分ですること**で最初は作るのが難しいだろうと思っていた作品でも完成させることができ達成感を感じました。
- プログラムを自分で作るにあたって、ChatGPTをはじめて使用してみたのですが、**完璧ではないコードの提案も多く、試行錯誤しました。**だんだん、**ChatGPTに修正をお願いするよりも、自分で座標とか角度とか変えたほうが早いということに気づいて、**数学の円の半径とか中心の知識と関連してるな、とか思いながら修正していて、楽しかったです。



## 個人的なまとめ

- 生成AIの活用によって**自己解決**できる生徒が増えた印象
- 「生成AIを使用した」 ≠ 「良いプログラム」
- 生成AIの使用の**判断**は生徒に委ねたい（強要はしない）
- 生成AIは生徒の**思考のサポート**の位置づけ
- 生成AIはプログラミング以外にも**有効なツール**となる

