この教材は『手を動かしながらやさしく学べるはじめてのAIデータサイエンスリテラシー』（技術評論社）の実践編として本書の v〜vii ページで紹介している発展的な実践教材です。本書で学んだことを活かして、発展的で面白いプロジェクトに挑戦できます。この教材はPDFで提供しています。必要に応じて印刷し、Wolframノートブックに自分でコードを入力して実行してください。（Wolframノートブックでの配布はしていません。）
なお、ここでは、本書独自のフォーマットを使い、プログラムの入力部分にオレンジの枠，出力部分にブルーの枠をつけてプログラムの部分をわかりやすく表示しています。新規でノートブックを開いた時にはこれらの色枠は付きません。

実践編３：Wolfram言語でアートプログラミングに挑戦しよう！

ランダム関数を使うと、同じプログラムコードでも、プログラムを実行するたびに違った結果が出てきます。この演習では、円を描く関数とランダム関数を使って、大小さまざまな円を組み合わせたアート作品を、Wolfram言語のプログラミングで作っていきましょう。

手順は以下の通りです。

1. Circle関数で円を描こう
2. 中心座標と半径にランダム関数を指定して円を描こう
3. 複数のランダムな円をまとめて描こう
4. 円に色をつけて完成！
5. 応用：円以外のグラフィックスや、乱数を使わないプログラムの例

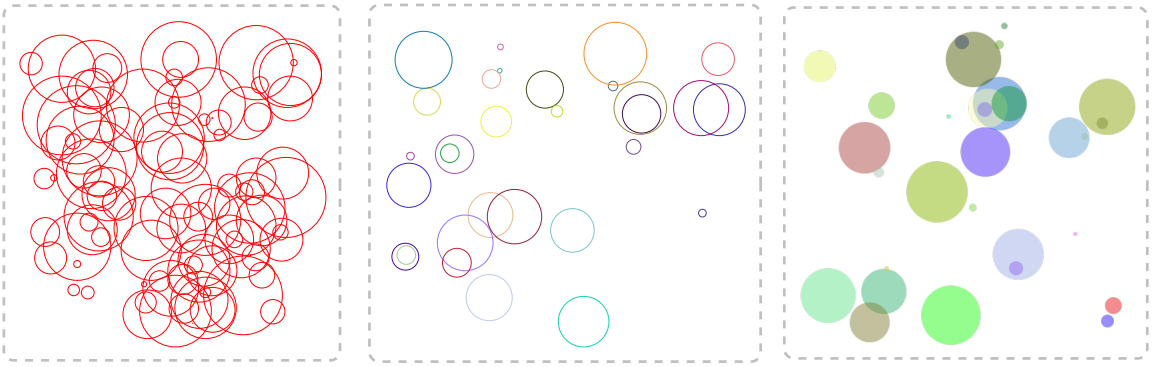


図1 アートプログラミングの作品例

1. Circle関数で円を描こう

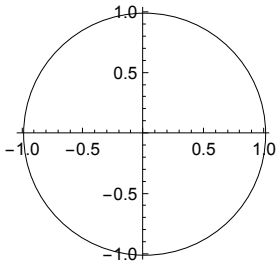
まず、**Circle**関数を使って円を描きましょう。

Graphics[Circle[{0,0},1],Axes→True] は『中心座標{0,0}で半径1の円を座標軸と共に描画する』というプログラムです。

＊注意＊
以下のプログラムで、→（矢印）は、マイナス記号と＞記号を続けて入力します。すると自動的に→に変わります。キーボードの矢印キーではありませんので注意してください。）

In[1]:=
```
Graphics[Circle[{0, 0}, 1], Axes → True]
```
[グラフ…　[円　[軸…　[真

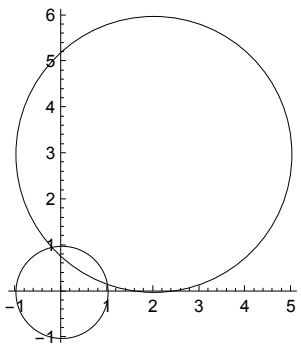
Out[1]=



複数の円を同時に描きたいときは、次のように**Circle**関数のプログラムをカンマで並べて{ } に入れ、円のリストを作ります。リストとは、 { } で複数の要素をひとつにまとめたものです。
以下の例では、中心{0,0}で半径1の円と、中心{2,3}で半径3の円を描いたものです。

In[2]:=
```
Graphics[{Circle[{0, 0}, 1], Circle[{2, 3}, 3]}, Axes → True]
```
[グラフィ…　[円　[円　[軸…　[真

Out[2]=



練習問題(1)：中心座標や半径の大きさを変えて、プログラムを実行（SHIFT+ENTER ）してみましょう。

2. 中心座標と半径にランダム関数を指定して円を描こう

次に、**RandomReal** 関数を組み合わせて、ランダムに選ばれた中心座標と半径で円を描いていきます。（**Real**は実数という意味です）

まずは、**RandomReal[10]**として、0から10の範囲の実数の乱数を生成してみましょう。プログラムを実行するたびに数値が変わります。何度も実行して数値が変わるか試してみましょう。

In[3]:=
```
RandomReal[10]
|実数乱数
```

Out[3]=
```
3.43002
```

RandomReal[10,2]とすると、0から10の範囲の数値の中から、2つの乱数がリスト**{*数値1*, *数値2*}**という形で出力されます。これも、実行するたびに数値が変わります。試してみましょう。

In[4]:=
```
RandomReal[10, 2]
|実数乱数
```

Out[4]=
```
{5.11817, 7.51461}
```

これらを**Circle**関数の引数として使用すれば、座標と半径をランダムに決めるプログラムになります。実際にどんな値になっているのかを、以下のプログラムを実行して確認してみましょう。

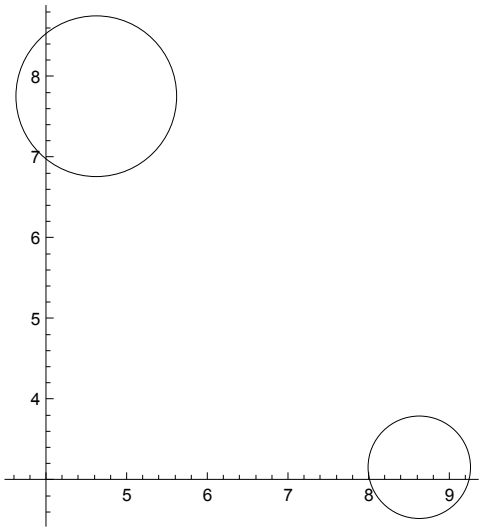
In[5]:=
```
Circle[RandomReal[10, 2], RandomReal[1]]
|円        |実数乱数                |実数乱数
```

Out[5]=
```
Circle[{6.52001, 5.15458}, 0.0565756]
```

それでは、上記1の最後のプログラムを少し変更して、ランダム関数を使ったプログラムにします。以下は、中心座標が{-10,-10}から{10,10}の範囲、半径が0から1の範囲の円を2つ描くプログラムです。プログラムを何度か実行し、実行するたびに描画される円が変わることを確認しましょう。

In[6]:=
```
Graphics[
 {Circle[RandomReal[10, 2], RandomReal[1]],
  Circle[RandomReal[10, 2], RandomReal[1]]},
 Axes → True]
```

Out[6]=



練習問題(2)：上のプログラムは、中心座標が{-10,-10}から{10,10}の範囲、半径が0から1の範囲です。この範囲を変えるにはどの数字を変えたらいいか考えて試してみましょう。

3. 複数のランダムな円をまとめて描こう

指示した内容を繰り返し何回でも実行することは、プログラムの得意技です。

まずは手始めに、次のプログラムを実行してみましょう。**abc**が5個のリストが生成されます。

In[7]:=
```
Table[abc, {5}]
```

Out[7]=
```
{abc, abc, abc, abc, abc}
```

次に、**abc**を**RandomReal[10]**に変えて実行してみましょう。0から10の範囲でランダムに選ばれた5個の数値のリストが生成されます。

In[8]:=
```
Table[RandomReal[10], {5}]
```

Out[8]=
```
{4.53609, 3.46122, 7.50842, 0.160085, 2.74014}
```

この**Table**のプログラムを、円を複数作るプログラムに変更していきます。ちょっと長くなりますが、次のプログラムでは、上記２でのランダムな中心座標とランダムな半径で、５個の円を作っています。

In[9]:=
```
Table[Circle[RandomReal[10, 2], RandomReal[1]], {5}]
```
リ… 円 実数乱数 実数乱数

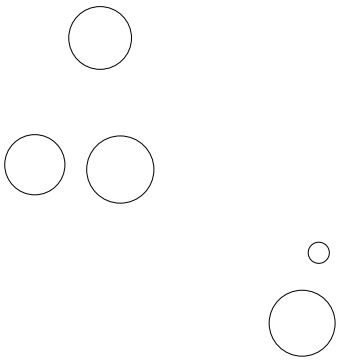
Out[9]=
```
{Circle[{9.65873, 4.26563}, 0.982738],
 Circle[{6.04585, 9.56051}, 0.349797], Circle[{1.76551, 6.10242}, 0.688778],
 Circle[{8.68382, 0.823234}, 0.364676], Circle[{2.14623, 0.889938}, 0.15913]}
```

これを描画すると、ランダムな位置と大きさで異なる5つの円が描けそうですね。

上のプログラムを**Graphics[]**の中にいれて、グラフィックスとして描画しましょう。

In[10]:=
```
Graphics[Table[Circle[RandomReal[10, 2], RandomReal[1]], {5}]]
```
グラフ… リ… 円 実数乱数 実数乱数

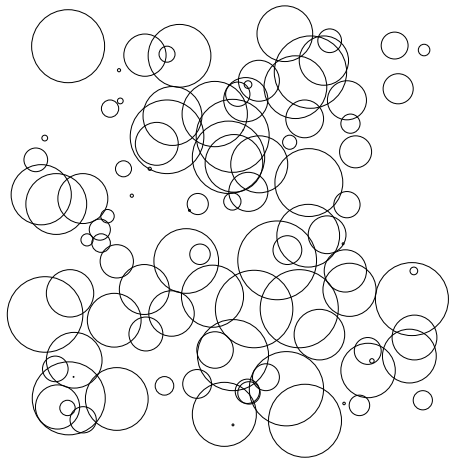
Out[10]=



円を増やして、100個描いてみます。よりアートっぽくなりますね！

In[11]:=
```
Graphics[Table[Circle[RandomReal[10, 2], RandomReal[1]], {100}]]
```
グラフ… リ… 円 実数乱数 実数乱数

Out[11]=



練習問題(3)：円の個数を変えたり、中心座標の範囲や半径の大きさの範囲を変えて、プログラムを実行してみましょう。

4. 円に色をつけて完成！

さあ、いよいよ、きれいな色をつけて、素敵なグラフィックスにしていきましょう。

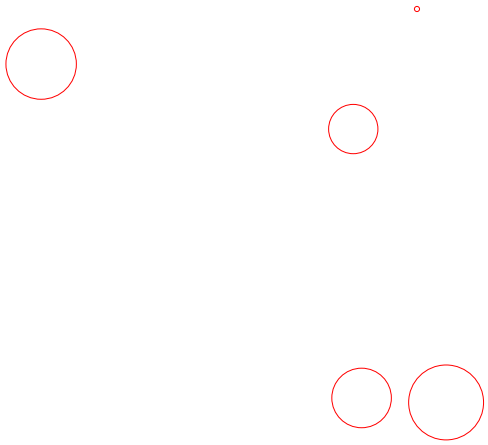
Circle関数の前に色の関数**Red**を追加し、

```
Red, Circle[RandomReal[10,2],RandomReal[1]]
```

を { } で括ると、「赤色で、半径が0から1の値をランダムに取る円が5個」描画されます。

In[12]:=
```
Graphics[Table[{Red, Circle[RandomReal[10, 2], RandomReal[1]]}, {5}]]
```
|グラフ… |リス… |赤 |円 |実数乱数 |実数乱数

Out[12]=

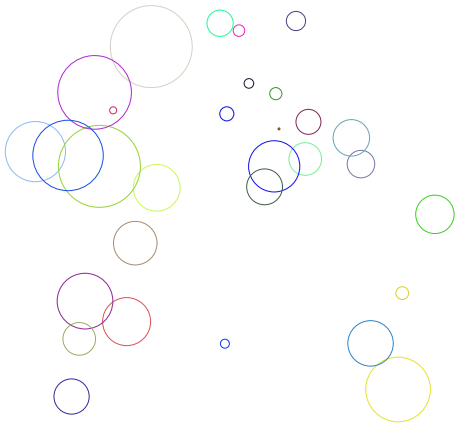


RandomColor関数を使えば、色もランダムにつけられます。

Redを**RandomColor[]** に変更すると、以下のようなカラフルな作品になります。

In[13]:=
```
Graphics[Table[{RandomColor[], Circle[RandomRealRandomReal[10, 2], RandomReal[1]]}, {30}]]
```
|グラフ… |リス… |ランダムな色 |円 |実数乱数 |実数乱数

Out[13]=



練習問題(4)：
円の個数や色などを自由に変えてみて、お気に入りの作品を作りましょう。第３章で学んだRGBColor関数や、実践編２で紹介した色関数を使うと色を指定できます。ランダムな関数を使うと、同じプログラムでも実行するたびに違った結果になります。

5. 応用：他の形のアート作品にチャレンジしてみよう

円を塗りつぶして円板にしたり、三角形にしたり、パラメータを規則的に動したりして、さまざまなグラフィックスを描画してオリジナルのアート作品を作りましょう。以下にいくつかの例を紹介します。

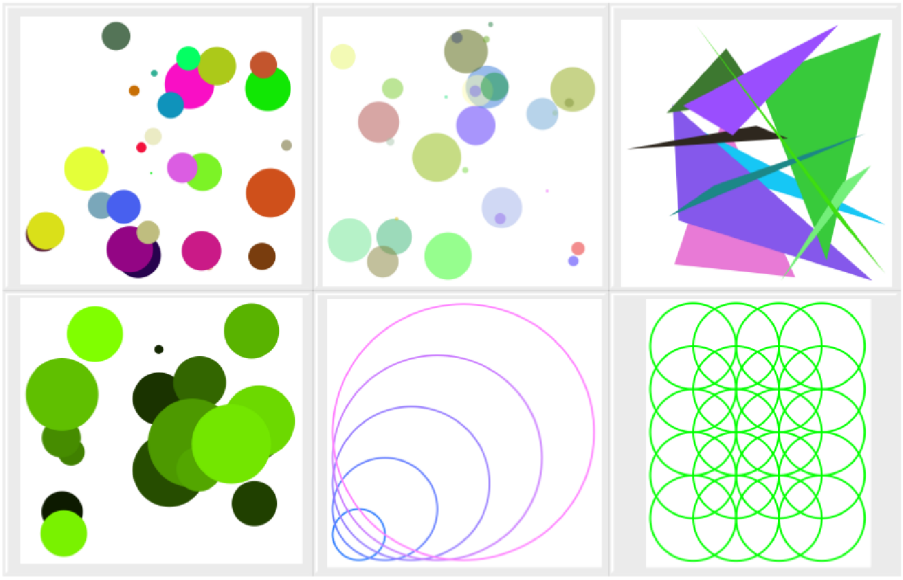


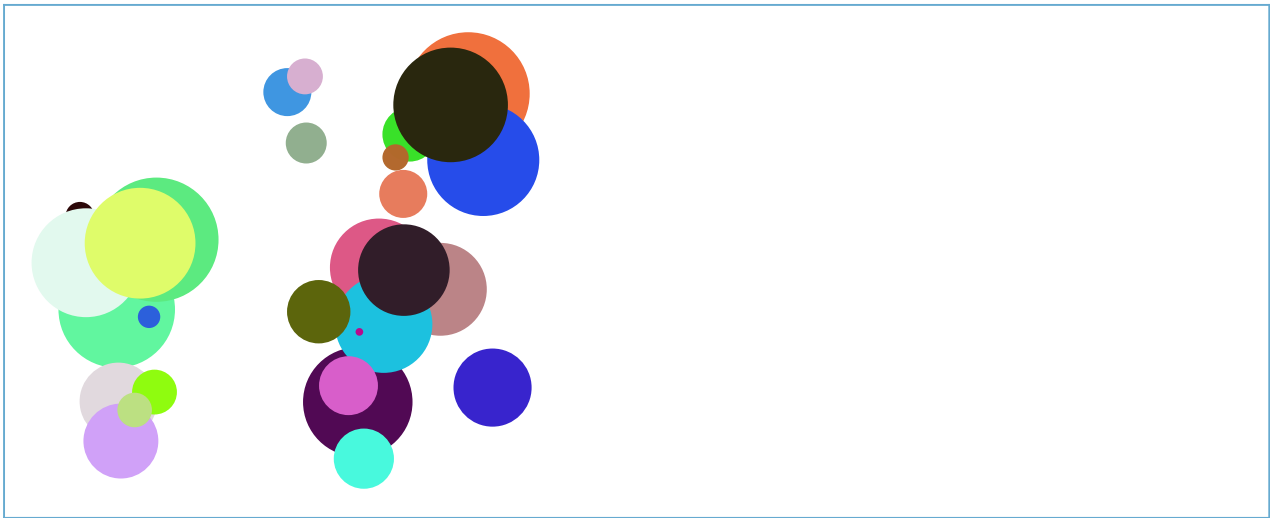
図2 アートプログラミングの応用作品の例

Wolfram言語の関数は、本書28ページのコラムで紹介した「オンラインドキュメント」で調べたり、例題を見たりすることができます。新しく出てきた関数は自分でドキュメントを調べて、いろいろ試してみてください。

5-1.CircleをDiskにする

In[14]:=
```
Graphics[Table[{RandomColor[], Disk[RandomReal[10, 2], RandomReal[1.5]]}, {30}]]
```
[グラフ… [リス… [ランダムな色 [円板 [実数乱数 [実数乱数

Out[14]=



5-2.Opacity関数を追加して透明度を加える

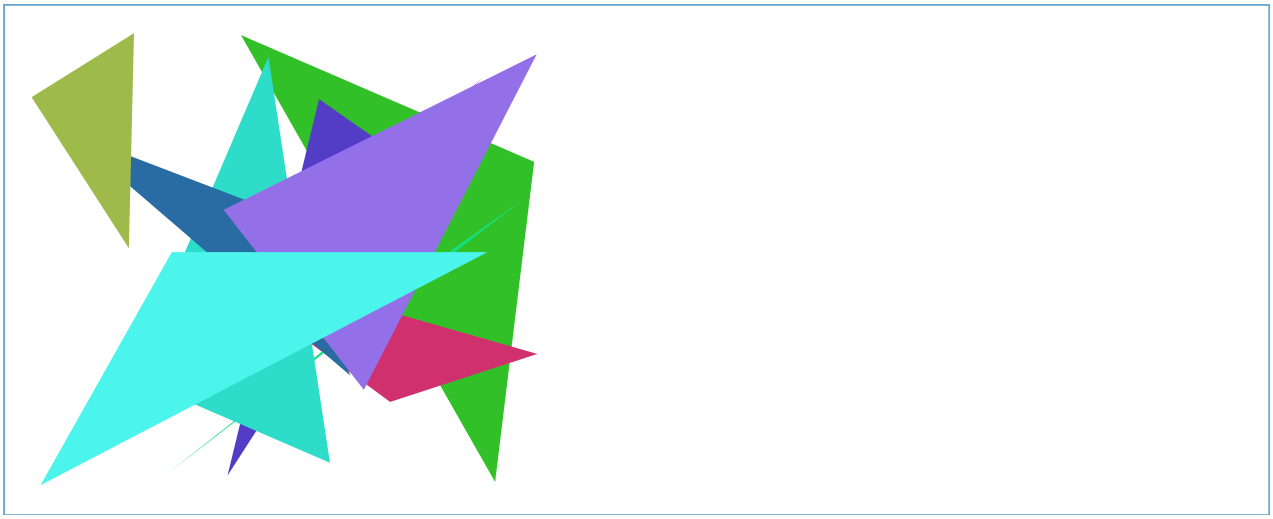
In[15]:=
```
Graphics[Table[
 {Opacity[0.7], RandomColor[], Disk[RandomReal[10, 2], RandomReal[1.5]]}, {30}]]
```

Out[15]=

5-3.Polygonで三角形をつくる

In[16]:=
```
Graphics[Table[{RandomColor[],
  Polygon[{RandomReal[10, 2], RandomReal[10, 2], RandomReal[10, 2]}]}, {10}]]
```

Out[16]=



ランダムではなく、規則的な変化をつけたいとき

Table関数を使って、パラメータの値が規則的に変わることで異なる色や配置ができるアートを作ることもできます。

5-4.RGBColor関数にパラメータ iを設定して、iの値を変えることで色を規則的に変化させる

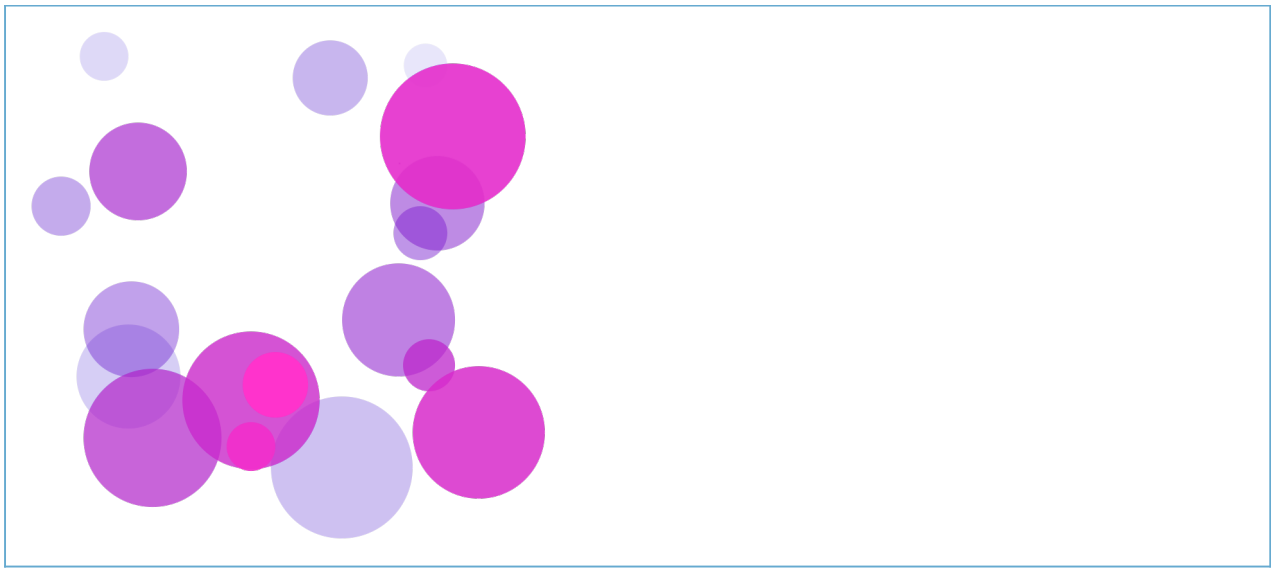
In[17]:=
```
Graphics[Table[{RGBColor[i, 0.2 i, 0.8],
  Disk[RandomReal[10, 2], RandomReal[1.5]]}, {i, 0.1, 1, 0.05}]]
```

Out[17]=

5-5.さらに透明度にもパラメータをつけて変化させる

In[18]:=
```
Graphics[Table[{Opacity[i], RGBColor[i, 0.2 i, 0.8],
  Disk[RandomReal[10, 2], RandomReal[1.5]]}, {i, 0.1, 1, 0.05}]]
```

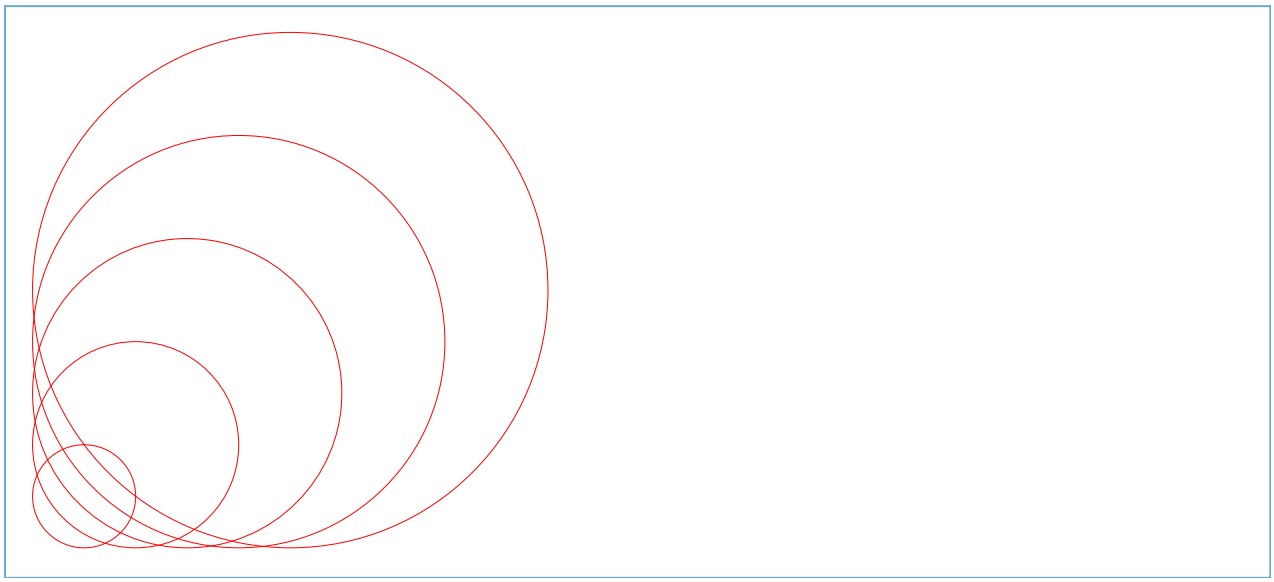
Out[18]=



5-6.Circle関数の円の中心と半径にパラメータを設定して変化させる

In[19]:=
```
Show[Table[Graphics[{RGBColor[1, 0, 0], Circle[{i, i + 2}, i]}], {i, 1, 5}]]
```
示す　リ…　グラフィ…　RGBカラー　円

Out[19]=



5-7.パラメータを2つ使って、中心座標を変化させる

In[20]:=
```
Show[Table[Graphics[{Blue, Circle[{m, n}, 1]}], {m, 1, 5}, {n, -2, 4}]]
```
示す　リ…　グラフィ…　青　円

Out[20]=