

Python100問チャレンジ

問題の解き方とアドバイス

Python100問チャレンジは、あなたのPythonやプログラミングに対する理解度を知り、高めてもらうために作りました。Chapter02からChapter11までについて、読んだ後に理解度の確認をするために解いてみてください。問題の最後にあるChapterXは総合問題として、プログラムの間違いを修正する問題に取り組んでいただきます。

難易度は3段階あります。難易度1(★)は、本書を読んで最低限押さえてほしい問題です。難易度2(★★)は、内容を咀嚼して考える力が求められる、小さなプログラムの動きについて確認する問題です。難易度3(★★★)は、プログラムを読む力が問われる問題です。難易度1の問題については、間違いが気になったり、手こずるようなら、もう一度本文を読み直してみてください。難易度2(★★)、難易度3(★★★)の問題は、解けなくてもあまり気にしないでください。

すべての問題に共通しているのは、問題文や選択肢の意味を考えることでより正解に近づける点です。文章をよく読んで、あなたの頭で考えて解答を導き出してください。

中には、プログラムを動かすことで解答を得られるタイプの問題もあります。そのような問題は、可能なら頭の中でプログラムを動かしてみても解いてください。正解かどうかを確かめるためにプログラムを書いて動かしてみてもよいでしょう。その際は、結果が出る理由についてよく考えてみるようにしてください。

また問題の解答だけでなく、解説もぜひ読んでみてください。正答した場合も誤答した場合も、問題のとらえ方、考え方のヒントを得られると思います。そして、本書で学んだ内容を一步深めることができるはずです。

解答と解説は、問題の後にまとめてあります。

では、楽しみながら頑張ってください！

100問チャレンジ 問題

Chapter 02

Q 2-1 難易度 ★☆☆

商品の値段に税率をかけて価格を計算するプログラムを作ろうと思います。商品の値段を代入するための変数名として、ふさわしいものを1つ選択肢から選んでください。

- a. x
- b. price
- c. a
- d. tax_rate

Q 2-2 難易度 ★☆☆

あるプログラムを読むと「book_price」という変数が使われていました。どんな種類のプログラムだと想像できるでしょうか。選択肢から適切なものを1つ選んでください。

- a. 図形の面積を計算する
- b. 書籍の価格を計算する
- c. 平均気温を計算する

Q 2-3 難易度 ★☆☆

Pythonの数値と演算について、間違っていることを次からすべて選び記号で教えてください。

- a. 小数点として「.」を使います
- b. 掛け算と割り算にはそれぞれ「*」「/」という記号を使います
- c. 計算式に「{ }」のような波括弧があると、先に計算されます
- d. 累乗を計算するためには「^」という記号を使います

Q 2-4 難易度 ★☆☆

次の中から、Pythonの文字列をすべて選び、記号で答えてください。

- a. リンゴ
- b. "orange"
- c. 125
- d. "3250"

Q 2-5 難易度 ★☆☆

Pythonのプログラムで計算をするとき、次の選択肢からエラーにならないものをすべて選んでください。

- a. $10 + 20$
- b. $10 + "20"$
- c. $"10" + "20"$
- d. $"0" * 100$

Q 2-6 難易度 ★☆☆

次の選択肢にあるaからdの4つの比較演算子と1から5の選択肢で、意味について、正しい組み合わせを選んでください。「aと1」「bと2」のように、組み合わせを答えてください。

- a. `==`
 - b. `>`
 - c. `>=`
 - d. `!=`
-
- 1. 左右が等しくない
 - 2. 左側が右以上
 - 3. 左側が右より大きい
 - 4. 左右が等しい
 - 5. 左側が右未満

Q 2-7 難易度 ★★★

次のプログラムを動かして、出力セルに「ゆっくり休みましょう」と表示されるのは、1行目の変数temperatureに体温としてどんな数値が代入されているときでしょうか(空欄部分)。条件に合う数値を、選択肢からすべて選んでください。

```
temperature =   
if temperature >= 37:  
    print("ゆっくり休みましょう")  
else:  
    print("平熱です")
```

- a. 100
- b. 36
- c. 37
- d. 23

Q 2-8 難易度 ★★★

次の文章にある**1**から**3**までの空欄にふさわしいものを**a**から**d**の選択肢から選んで、「1:a」「2:b」のように答えてください。

Pythonの 1 につける名前を 1 名といいます。 2 のような名前をつけてもエラーにはなりませんが、 3 のような名前をつけて、どんな種類の数値やデータが代入されているかがよくわかるように心がけましょう。

- a. priceやarea
- b. 変数
- c. xやa
- d. 代入

Q 2-9 難易度 ★★☆☆

次の文章にある1から3までの空欄にふさわしいものをaからdの選択肢から選んで、「1:a」「2:b」のように解答してください。

関数を定義するとき、関数名に続けて書く丸括弧の中に変数のようなものを置くことができます。これを 1 といいます。 1 は関数のブロック内で計算に使われます。関数を 2 ときは、関数名に続けて関数に与えたい数値などを書きます。そうすることで、データが 1 に受け渡され、計算に使われて結果が 3 として返ってきます。これが関数の動く仕組みです。

- a. 引数
- b. 戻り値
- c. 入力
- d. 呼び出す

Q 2-10 難易度 ★★☆☆

次の組み込み関数を呼び出したとき、戻り値が数値の100になるものをすべて選んでください。

- a. `int("100")`
- b. `str(100)`
- c. `min(0, 100)`
- d. `pow(100, 1)`
- e. `max(100, 0)`

Q 2-11 難易度 ★★★

次に示すのは、平行四辺形の面積を計算するプログラムを、行ごとに順番を入れ替えたものです。並べ替えて正しく動く1つのプログラムにしたとき、一番最後にくる行を記号で答えてください。

- a. height = 5 # 高さ
- b. area = base * height
- c. base = 4 # 底辺

Q 2-12 難易度 ★★★

Q2-11の平行四辺形の面積計算プログラムを関数にしてみました。1から3までの空欄にふさわしい選択肢を選んで、「1: a」「2: b」のように解答してください。

```
1 calc_parallelogram_area(base, height):  
  2 = base * height  
  3 area
```

- a. return
- b. area
- c. lenth
- d. def

Chapter 03

Q 3-1 難易度 ★☆☆

あるテストの点数を4人分、「[75, 68, 93, 84]」のようなリストにして、pointsという変数に代入してとします。角括弧を使って「1から数えて3番目」の要素を取り出したいとき、インデックスとしてどのような数値を与えればよいでしょうか。選択肢から1つ選んでください。

- a. points[3]
- b. points[0]
- c. points[2]

Q 3-2 難易度 ★☆☆

次の1から3までの3種類の辞書があります。それぞれがプログラム上で扱おうとしているものとして適切なものを、選択肢aからdの中から選んでください。「1:a」「2:b」のように組み合わせで解答してください。

1. {"name": "田中太郎", "address": "東京都千代田区", "phone": "070-000"}
 2. {"和名": "ヒト", "種": "Homo sapience", "類": "霊長類"}
 3. {"temperature": 22.5, "humidity": 55.1, "pressure": 1013.2}
- a. 県庁所在地の位置情報
 - b. 気象の情報
 - c. アドレス帳アプリの項目
 - d. 生物の分類

Q 3-3 難易度 ★☆☆

「{"和名": "ヒト", "種": "Homo sapience", "類": "霊長類"}」という辞書がhitoという変数に代入されているとします。辞書に角括弧を添えて「霊長類」という値を取り出したいとき、どのように書けばいいでしょうか。選択肢の中から適切なものを1つ選んで答えてください。

- a. hito[2]
- b. hito["類"]
- c. hito.類
- d. hito[類]

Q 3-4 難易度 ★☆☆

次の選択肢にある解説の中で、リスト、辞書、タプルのうちタプルのみについて成り立つものをすべて選んで答えてください。

- a. 角括弧 ([]) で要素にアクセスする
- b. 要素の置き換えができない
- c. インデックスを使って要素を管理する
- d. キーを使って要素を管理する

Q 3-5 難易度 ★☆☆

for文を使ったループでよく使うrange()についての問題です。「range(5)」のようになると、0から4まで順番に増えるループを実行できます。次の1から3までのrange()の呼び出し方と、選択肢aからfまでの数の増え方をよく見て、1から3との組み合わせで正しいものを、「1:a」「2:b」のように答えてください。

1. range(10)
 2. range(5, 10)
 3. range(2, 3)
-
- a. 2から3まで
 - b. 5から9まで
 - c. 5から10まで
 - d. 0から9まで
 - e. 2のみ
 - f. 0から10まで

Q 3-6 難易度 ★☆☆

次の文章にある1から3までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように選択肢の記号を答えてください。

Pythonで使えるデータ構造のうち、Chapter03で紹介した3種類について、簡単に特徴を解説します。 1 は、同じ種類のデータを並べるのに使います。 2 は、異なる種類のデータを組み合わせたデータ構造に使い、キーと値のペアを変数名と代入されるデータのよう

に組み合わせて使います。 3 は 1 ととてもよく似たデータ構造ですが、異なった種類のデータを扱うのに使うため、名前のない 2 のような特徴を持っています。

- a. タプル
- b. 文字列
- c. リスト
- d. 辞書

Q 3-7 難易度 ★★☆☆

「{"temperature": 22.5, "humidity": 55.1, "pressure": 1013.2}」という辞書が weather という変数に代入されているとします。このとき、プログラムとして実行してエラーにならないものをすべて選んでください。

- a. weather["wind_speed"] = 3.5
- b. temp = weather["temp"]
- c. weather["pressure"] = 956.2
- d. print(weather["temperature"])

Q 3-8 難易度 ★★☆☆

あるテストの点数を4人分、「[75, 68, 93, 84]」のリストにして、points という変数に代入してあるとします。このとき、プログラムとしてエラーにならないものを選択肢からすべて選んでください。

- a. points[4]
- b. points[0]
- c. points[len(points)]
- d. points[3] = 100

Q 3-9 難易度 ★★☆☆

次のようなプログラムを実行したとき、出力セルにどのような結果が表示されるでしょうか。選択肢から選んでください。

```
human = {"和名": "ヒト", "種": "Homo sapience", "類": "霊長類"}
for category in human:
    print(category)
```

- a. 「ヒト」「Homo sapience」「霊長類」の順番で縦に表示される
- b. 「和名」「種」「類」の順番で縦に表示される
- c. 「類」「種」「和名」の順番で縦に表示される

Q 3-10 難易度 ★★★

少し頭の体操をしましょう。「[1, 2, 3, 10]」というリストをnumsという変数に代入しました。このとき、出力セルに10という答えが表示されるプログラムをすべて選んでください。

- a. `(nums[2] + nums[3]) * nums[2]`
- b. `nums[3]`
- c. `nums[3] / 2 * nums[2]`
- d. `nums[4]`
- e. `nums[1] * (nums[1] + nums[2])`

Q 3-11 難易度 ★★★

点数のリストを使って、合計を計算するプログラムを作りたいと思います。forを使ったループを使います。1から3の3つある空欄に入るものを選択肢から選んで、「1:a」「2:b」のように答えてください。

```
points = [75, 68, 93, 84] # 点数のリスト
1 = 0 # 合計を足していく変数を初期化
for point 2 points:
    1 = total + 3
total # 合計を確認
```

- a. `def`
- b. `p`
- c. `in`
- d. `point`
- e. `total`

Q 3-12 難易度 ★★★

インポートされている関数についての解説を読み、プログラムを動かしたときにどのような結果が出力セルに表示されるかについて、選択肢から正しい記号を選んで答えてください。

・関数の解説

random(): 0以上1未満の小数をランダムに返す関数です。

```
from random import random # random関数をインポート
eye = random() * 6 # さいころの目
print(eye)
```

- a. 1以上6未満のランダムな小数
- b. 0以上6未満のランダムな小数
- c. 0以上5未満のランダムな小数
- d. 0以上1未満のランダムな小数

Chapter 04

Q 4-1 難易度 ★☆☆

次の選択肢の中から、数値リテラルとして扱われるものをすべて選んでください。

- a. "1024"
- b. 0xFFE0
- c. 4096
- d. [10, 20, 30]

Q 4-2 難易度 ★☆☆

次の中から、文字列リテラルではないものをすべて選んで記号で答えてください。

- a. `"""Python"""`
- b. `"JavaScript"`
- c. `"Ruby¥nPHP"`
- d. `Go`
- e. `"200"`

Q 4-3 難易度 ★☆☆

次の選択肢にあるリテラルのうち、タプルとして扱われるものをすべて選んで記号で答えてください。

- a. `[1, 2, 3]`
- b. `(1, 2, 3)`
- c. `"Python"`
- d. `"Java", "Ruby", "Haskell"`

Q 4-4 難易度 ★☆☆

リスト型のメソッドには、戻り値を返すものと返さないものがあります。次の選択肢にあるリスト型メソッドのうち、戻り値を返さないものをすべて選んで記号で答えてください。「破壊的操作」という言葉を思い出して、またメソッド名の意味を考えて選んでください。「L」は任意のリスト型のオブジェクトを指すものとします。

- a. `L.count()`
- b. `L.remove()`
- c. `L.index()`
- d. `L.clear()`

Q 4-5 難易度 ★★☆☆

次の文章にある1から3までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

プログラムで扱うデータにはいろいろな種類があります。この種類のことを 1 と呼びます。Pythonではプログラムを作るときに基本となる 1 を 2 として分類していて、いつでも使えるようになっています。2 には数値型、文字列型やリスト型などいろいろな種類があり、それぞれ決められた記法である 3 を使ってデータを記述できるようになっています。

- a. 複素数
- b. 組み込み型
- c. リテラル
- d. 型

Q 4-6 難易度 ★★☆☆

次の文章にある1から3までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

「quote」という変数に「abc」という文字列が代入されているとします。このとき「quote.capitalize()」のようなプログラムを書くことで、英字を大文字にした「ABC」という文字列を戻り値として得ることができます。「quote」の部分は 1 と呼ばれ、文章にしたとき主語に相当します。「capitalize」は動詞で、2 と呼ばれます。このようにプログラムを作っていく手法には 3 という名前がつけられています。Pythonの組み込み型は 1 として振る舞います。

- a. アトリビュート
- b. オブジェクト指向
- c. オブジェクト
- d. メソッド

Q 4-7 難易度 ★★☆☆

次の選択肢には、if文に添える条件式だけを抜き出してあります。各条件式をif文に添えたとき、直後のブロックが実行される(条件式がTrueと判断される)ものをすべて選んで記号で答えてください。

- a. ".png" in "image.jpg"
- b. "20" > "100"
- c. "PYTHON" == "PYTHON"
- d. int("20") > int("100")

Q 4-8 難易度 ★★☆☆

a_listという変数に「[1, 2, 3, 4, 5]」というリストが代入されているものとします。このとき、**1**から**4**のプログラムが出力として返す正しい組み合わせを選択肢**a**から**e**の中から選んで、「1:a」「2:b」のように答えてください。

- 1. a_list[0:2]
- 2. a_list[0]
- 3. a_list[0:1]
- 4. a_list[:2] + a_list[2:]
- a. [1, 2]
- b. [1]
- c. [1, 2, 3, 4, 5]
- d. 1
- e. [1, 2, 3]

Q 4-9 難易度 ★★☆☆

変数のlstには任意のリストが、変数tplには任意のタプルが代入されているものとします。要素数はどちらも5です。このとき、エラーにならないプログラムを選択肢からすべて選んで記号で答えてください。

- a. `lst[1:3] = [1]`
- b. `tpl[4] = 5`
- c. `tpl.clear()`
- d. `tpl.count(0)`

Q 4-10 難易度 ★★☆☆

次の選択肢のプログラムは、リストを使ったプログラムの行を入れ替えたものです。リストを降順(数値の大きい順)に並べ替えるとき、一番最後に実行すべき行の記号を答えてください。

- a. `lons.reverse()`
- b. `lons = [35.18, 40.82, 34.39, 35.44]`
- c. `lons.sort()`

Chapter 05

Q 5-1 難易度 ★☆☆

次の選択肢の中から、辞書のキーについての説明として正しいものをすべて選んで記号で答えてください。

- a. キーとして数値や文字列を使えるが、リストは使えない
- b. 同じ辞書に重複したキーを登録できる
- c. キーの登録順は保存される
- d. 辞書のキーとしてタプル型が使える

Q 5-2 難易度 ★☆☆

「{1: "I", 2: "II", 3: "III"}」という辞書がroman_dicという変数に代入されているとします。この辞書に4というキーが存在しているかどうかをif文で調べたいとき、条件式はどのように書けばいいでしょうか。選択肢から正しいものを1つ選んで記号で教えてください。

- a. roman_dic[4] == 0
- b. roman_dic in 4
- c. 4 in roman_dic
- d. roman_dic > 4

Q 5-3 難易度 ★☆☆

次の選択肢のうち、シーケンス型の説明として正しい選択肢をすべて選んで記号を教えてください。

- a. 複数の要素を持つ
- b. 順番がない
- c. for文のinの右側に添えることができる
- d. 角括弧で要素にアクセスできる
- e. 要素の重複を許さない

Q 5-4 難易度 ★☆☆

次の選択肢のうち、イミュータブルな組み込み型の説明として正しい選択肢をすべて選んで記号を教えてください。

- a. 要素の変更ができる
- b. 辞書のキーになることができる
- c. set型の要素になることができない
- d. 破壊的操作を実行できる

Q 5-5 難易度 ★★☆☆

次の文章にある1から3までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

辞書型は 1 と 2 をペアにして登録することでデータ構造としてプログラムで使えます。 2 にはさまざまな組み込み型を登録できます。 1 として登録できる組み込み型は限られていて、文字列型、数値型のように 3 できない組み込み型だけを使うことができます。リストのように 3 できるものは 1 にすることはできません。

- a. タプル
- b. 値
- c. 変更
- d. キー

Q 5-6 難易度 ★★☆☆

次の文章にある1から3までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

Pythonの組み込み型には文字を扱うための型がいくつかあります。 1 は 2 済みの文字データを扱うための型です。 3 は 2 される前の文字データを扱うための型です。 1 は一般的な文字を扱うのに使われるのに対して、 3 はインターネットから取り出した直後の文字データや「バイナリデータ」と呼ばれる文字以外の「生のデータ」を扱うために使われます。Pythonの外部からやってくるデータは 3 として取り出されることがあり、文字として扱うためには 2 を行って 1 に変換するようにするとプログラムが作りやすくなります。

- a. bytes型
- b. エンコード
- c. 文字列型
- d. ユニコード

Q 5-7 難易度 ★★☆☆

{1: "I", 2: "II", 3: "III"}という辞書がroman_dicという変数に代入されているものとします。次の選択肢にあるプログラムのうち、エラーにならないものをすべて選んで記号を答えてください。

- a. roman_dic[4]
- b. roman_dic.get(4, "IV")
- c. 4 in roman_dic
- d. roman_dic[4] = "IV"

Q 5-8 難易度 ★★☆☆

次の文章にある1から3までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

set型はリストのように複数の要素を持つことができるデータ型ですが、「1」という要素を持ったset型に「1」を 1 しようとしても 1 が行われません。重複した要素を持たないのが特徴で、数学の 2 のような性質を持っています。ただし、追加できる要素には制限があって、辞書のキーと同じように 3 できない組み込み型のみを追加できます。

- a. 集合
- b. 変更
- c. リテラル
- d. 追加

Q 5-9 難易度 ★★★

「1 → January」のように月の数値を英語の月名に変換するプログラムを作るため、辞書を活用しようと思います。テストとして、1月から3月までについて、month_numという変数に代入した数値から月の英語名を得る次のプログラムを作ってみました。3つある空欄に入るものを選択肢から選んで、「1:a」「2:b」のように答えてください。

```
1 = {1: "January", 2: "February", 3: "March"}
month_num = 1    # 月の数値
name = month_names. 2 (month_num, " 3 ")
```

- a. month_names
- b. items
- c. get
- d. 未登録です
- e. 登録済みです

Q 5-10 難易度 ★★★

Pythonのファイル処理は決まった順番に実行する必要があります。次の選択肢の項目を、ファイル処理を行う順番に並べ替えて、記号を「a→b→c」のように順番に並べてください。なお、「F」は任意のファイルオブジェクトとします。

- a. open()関数を使ってファイルを開く
- b. F.close()関数を使ってファイルを閉じる
- c. F.read()、F.write()を使ってファイルの操作をする

Chapter 06

Q 6-1 難易度 ★☆☆

次の選択肢の中から、Pythonの式についての説明としてふさわしいと思うものをすべて選んで、記号で答えてください。

- a. 必ず改行を伴う
- b. 値を持つ
- c. 代入やifなど
- d. 計算、関数呼び出しなど

Q 6-2 難易度 ★☆☆

次の比較演算子を使った式のうち、Trueとなりifに添えるとブロックが実行されるものをすべて選んで記号で答えてください。

- a. `(1, 2, 3) == [1, 2, 3]`
- b. `[2, 3, 4] == [2, 3, 4]`
- c. `"abc" == b"abc"`
- d. `100 == "100"`

Q 6-3 難易度 ★☆☆

次の式のうち、bool型(真偽値)として判定するとTrueになり、ifに添えるとブロックが実行されるものをすべて選んで記号で答えてください。

- a. `"Python"`
- b. `[]`
- c. `{1, 2, 3}`
- d. `[1, 2, 3][0:0]`

Q 6-4 難易度 ★☆☆

次の1から3の文法と説明の選択肢aからcで、Pythonのfor文で使う文法と説明の組み合わせを選んでください。「1:a」「2:b」のように組み合わせを答えてください。

1. break
 2. continue
 3. else
-
- a. ループブロックの先頭に戻る
 - b. ループが正常終了したときだけ以降のブロックを実行する
 - c. ループを終了する

Q 6-5 難易度 ★☆☆

次の文章にある1から3の空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

Pythonの代入文を 1 と 2 という言葉を通して眺め直してみましょう。=(イコール)の左右に 1 を置くのがPythonの代入 2 の文法です。右側には計算 1 、関数呼び出しのような 1 が置かれます。左側は代入という命令の対象となる変数名が置かれる他、コンマで区切った複数の変数をタプルとして置くと 3 代入になり、またリストのスライスを置いて代入で複数の要素を置き換えることができます。タプルもスライスも 1 であることには変わりはありません。プログラムの要素を 1 と 2 としてとらえることで、Pythonの文法をより抽象的に把握できるようになり、文法への理解が深まるのです。

- a. 文
- b. 複合演算子
- c. アンパック
- d. 式

Q 6-6 難易度 ★★☆☆

次の文章にある1から3までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

Pythonでは 1 文によって変数が作られますが、置かれる場所によって変数の性質が変わるようになっています。def文のブロック内で 1 が行われた変数は 2 変数となり、関数内でしか生存できません。def文のブロック外で 1 が行われると 3 変数となります。 3 変数は関数の外部と内部どちらからも見ることはできますが、 2 変数は関数内部からしか見ることはできない、という違いがあります。

- a. 宣言
- b. グローバル
- c. 代入
- d. ローカル

Q 6-7 難易度 ★★☆☆

次の文章にある1から3までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

1 も 2 ももともとは関数についてのドキュメンテーション(説明書き)をする目的を持った機能です。 1 はdef文の直下に三重引用符で囲まれた文字列として置かれます。これは改行を含む場合が多いからです。 2 は引数やdef文の末尾に、 3 を添えることができる文法です。引数や戻り値についての説明を文字列として置くこともできますが、intやstrのように型がわかるような関数名を置くことが慣習化しています。文字列も関数名もPythonでは値を持った 3 なので、 2 としてプログラム内に書くことができます。

- a. 呼び出し可能オブジェクト
- b. アノテーション
- c. 式
- d. ドックストリング



難易度 ★★☆☆

次のような関数`my_sum()`の機能についての説明として正しいものを**a**から**c**の選択肢から1つ選び記号で答えてください。

```
def my_sum(*args):  
    total = 0  
    for num in args:  
        total += num  
    return total
```

- a.** 数値(など)を要素として持つリストを受け取り、要素をすべて加算し結果を返す
- b.** コンマで区切った引数を受け取り、すべて加算して結果を返す。ただし1つ以上の引数が必要
- c.** コンマで区切った引数を受け取り、すべて加算して結果を返す。引数はゼロ個でもいい

Q 6-9

難易度 ★★☆☆

for文で書かれた「# 書き換えるプログラム」を、リスト内包表記を使って「# 書き換えたプログラム」に書き換えたいと思います。3つある空欄に入るものを選択肢から選んで、「1:a」「2:b」のように答えてください。

```
# 書き換えるプログラム
str_list = ["100", "200", "300"]
num_list = [] # 結果のリストを初期化
for num_str in str_list:
    num_list.append(int(num_str))
```

```
# 書き換えたプログラム
str_list = ["100", "200", "300"]
1 = [int( 2 ) for num_str in 3 ]
```

- a. num_str
- b. str_list
- c. num_list
- d. if

Q 6-10

難易度 ★★★

次のリスト内包表記のプログラムを動かした後、num_listにはどのような値が入っているでしょうか。正しいものをaからdの選択肢から選んでください。

```
str_list = ["100", "200.0", "300"]
num_list = [int(num_str) for num_str in str_list
↳         if "." not in num_str]
```

- a. [100, 200, 300]
- b. ["100", "200.0", "300"]
- c. [100, 300]
- d. [200.0]

Q 6-11 難易度 ★★★

次のように定義された関数があるとします。戻り値が「200」となるような呼び出し方を、選択肢から1つ選んで答えてください。

```
def calc_price_with_tax(price, tax_rate = 0.2):  
    total = price * (1 + tax_rate)  
    return round(total)
```

- a. calc_price_with_tax(200)
- b. calc_price_with_tax(180)
- c. calc_price_with_tax(200, 10)
- d. calc_price_with_tax(200, 0)

Q 6-12 難易度 ★★★

文字列のリストの項目を、「1: 数値型」のように1からはじまる番号を添えて順番に表示するプログラムを作りました。3つある空欄に入るものを選択肢から選んで、「1:a」「2:b」のように答えてください。

```
1 = ["数値型", "文字列型", "リスト型", "辞書型"]  
for 2, 3 in enumerate(builtins):  
    print(f"{num+1} : {name}")
```

- a. builtins
- b. range
- c. name
- d. num

Chapter 07

Q 7-1 難易度 ★☆☆

オブジェクトの説明として、本書の解説と合うものを選択肢の中からすべて選び記号で答えてください。

- a. データ構造を持っている
- b. 変数に代入できない
- c. アトリビュートを持っている
- d. クラスから作られる

Q 7-2 難易度 ★☆☆

本書の解説のアトリビュートの説明から考えて、選択肢で間違っているものをすべて選び、記号で答えてください。

- a. 変数のように名前(アトリビュート名)を持っている
- b. オブジェクトに続いてアンダースコア()を挟んで記述する
- c. 一度データを代入すると、二度以降は代入して変更できなくなる
- d. 数値だけでなく、いろいろな値を代入できる

Q 7-3 難易度 ★☆☆

Pythonで、クラスの定義をするために使われる文法として正しいものを選択肢の中から選び、記号で答えてください。

- a. object文
- b. def文
- c. class文
- d. for文

Q 7-4 難易度 ★☆☆

Pythonのクラスで、メソッドを定義するために使われる文法として正しいものを選択肢の中から選び、記号で答えてください。

- a. def文
- b. method文
- c. class文
- d. if文

Q 7-5 難易度 ★☆☆

Pythonのクラス定義で使われる2種類の名前が**1**と**2**にあります。この解説としてふさわしいものを**a**から**d**の選択肢から選んで、正しい組み合わせを「1:a」「2:b」のように答えてください。

- 1. `__init__`
 - 2. `self`
- a. メソッドの第1引数として使われる名前
 - b. クラス定義を空にしたいときに置かれる名前
 - c. オブジェクトが初期化されるときに呼ばれるメソッド名
 - d. クラス定義の終わりを示すときに置かれる名前

Q 7-6 難易度 ★☆☆

次のような、標準モジュールにあるdatetimeクラスを使うプログラムを作りました。このプログラムを動かしたとき、出力セルに表示される数値を答えてください。最後の行をよく見て答えてください。

```
from datetime import datetime
a_day = datetime(2100, 2, 1, 12, 0, 0)
# 2100年2月1日12時0分0秒
a_day.year
```

Q 7-7 難易度 ★★★

次の文章にある**1**から**3**までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

プログラムではさまざまな **1** を扱います。Pythonでは、変数の集まり、辞書など、いろいろな方法で **1** を扱うことができますが、とりわけ **2** を使ってプログラムを開発する手法を、**2** 指向プログラミングと呼んでいます。Pythonでは、**2** に紐づく **3** にデータを代入することで **1** を表現します。

- a. オブジェクト
- b. クラス
- c. アトリビュート
- d. データ構造

Q 7-8 難易度 ★★★

次の文章にある**1**から**3**までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

Pythonの **1** 定義では、class文から続くブロックに **1** の情報を定義します。ブロック内にdef文を置くことで定義される **2** のうち、__init__は特別な **2** につけられる名前です。他の **2** と同様、第1引数のselfにオブジェクトが渡されて初期化時に呼び出され、**2** 内でselfに対して **3** を追加するなどしてデータ構造としての体裁を整えます。**3** の初期値が必要なときは、self以降に引数を定義して受け取り **3** に代入します。初期化 **2** は、**1** がオブジェクトの設計図として振る舞うために重要な役割を担っているのです。

- a. データ構造
- b. メソッド
- c. クラス
- d. アトリビュート

Q 7-9 難易度 ★★★

次のように、name (商品名) と price (価格) をデータ構造として持つ Product クラスを定義しました。このクラスからオブジェクトを作る方法として正しいものを選択肢から選んでください。

```
class Product:
    def __init__(self, name, price):
        self.name = name
        self.price = price
```

- a. notebook = Product()
- b. pencil = Product("えんぴつ", 150)
- c. eraser = [Product, "消しゴム", 150]

Q 7-10 難易度 ★★★

長方形を表現するための簡易なクラス「Rectangle」をPythonで作りたいと思います。幅 (width)、高さ (height) をアトリビュートとして持ち、area() メソッドを使うことで戻り値として面積を返します。3つある空欄に入るものを選択肢から選んで、「1:a」「2:b」のように答えてください。

```
1 Rectangle:
2     __init__(self, width, height):
    self.width = width
    self.height = height

2     area(self):
    return self. 3 * self.height
```

- a. length
- b. def
- c. width
- d. new
- e. class

Chapter 08

Q 8-1 難易度 ★☆☆

次の選択肢から、Pythonの組み込み型の解説として間違っているものをすべて選んで記号で答えてください。

- a. 数値型はメソッドを持っていない
- b. リストや辞書で要素にアクセスするときは、内部でメソッドが呼ばれている
- c. 演算子を使った加算では、異なる組み込み型であっても内部では同じ名前のメソッドが呼ばれている
- d. 数値型を文字列型に変換するときは、文字列型のメソッドが呼ばれる

Q 8-2 難易度 ★☆☆

次の選択肢の中から、Pythonの特殊メソッドの説明として正しいものをすべて選んで記号で答えてください。

- a. メソッド名がアルファベットの太文字だけで構成されている
- b. 1つのアンダースコア(_)で前後が囲まれている
- c. 演算を行うときに暗黙に呼ばれる
- d. ダンダーメソッドとも呼ばれる

Q 8-3 難易度 ★☆☆

Colaboratoryのセルに、変数など値を持つ式だけを書いた行があるとき、値が文字列の場合はシングルクォーテーションで囲まれ、オブジェクトの場合は「datetime.date(2100, 2, 1)」のような文字列が出力セルに表示されます。この表示の名称を、選択肢から選んで記号で答えてください。

- a. 標準出力
- b. str
- c. 変換文字列
- d. repr

Q 8-4 難易度 ★☆☆

1から3の特殊メソッドと、選択肢aからcの機能の解説について、正しい組み合わせを選んでください。「1:a」「2:b」のように答えてください。

1. `__str__()`
2. `__len__()`
3. `__hash__()`

- a. オブジェクトのハッシュ値を計算して返します
- b. オブジェクトを文字列に変換します
- c. オブジェクトの要素数を返します

Q 8-5 難易度 ★☆☆

あるクラスの機能を拡張するなどして、親となるクラスを引き継ぐことを何と呼ぶでしょうか。適切なものを選択肢の中から1つ選んで記号で答えてください。

- a. オーバードライブ
- b. 継承
- c. 引き継ぎ
- d. 援用

Q 8-6 難易度 ★☆☆

Pythonのリスト型の機能を拡張するため、リスト型を親クラスとして子クラス「MyList」を作ろうと思います。このとき、クラス定義をどのように書けばいいでしょうか。適切なものを選択肢の中から1つ選んで記号で答えてください。

- a. `class MyList -> list:`
- b. `class list(MyList):`
- c. `class MyList(list):`
- d. `class MyList : list`

Q 8-7 難易度 ★★☆☆

Pythonの辞書型の機能を拡張するため、辞書型を親クラスとして子クラス「MyDict」を作ろうと思います。子クラス側に定義された特殊メソッドの挙動として、正しいものを選択肢の中から1つ選んで記号で答えてください。

- a. 子クラスのメソッドが呼び出された後、親クラスの同名のメソッドが自動的に呼び出される
- b. 親クラスのメソッドが呼び出された後、子クラスのメソッドが呼び出される
- c. 子クラスのメソッドは呼び出されず、親クラスのメソッドが呼び出される
- d. 子クラスのメソッドが呼び出され、親クラスのメソッドは呼び出されない

Q 8-8 難易度 ★★☆☆

次の文章にある**1**から**3**までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

Pythonでは演算や比較、インデックスを使った要素へのアクセス、演算子を使った計算などさまざまな場面で **1** が使われています。「+」演算子なら「__add__」、というように、同じ操作には同じ **1** 名が割りあてられています。組み込み型を含む **2** の基本的なふるまいのことを **3** と言いますが、これは **1** の有無によって決まっていると言えます。このような仕組みを知ると、**2** 指向をPythonがどのように実装しようとしているのかがとてもよくわかります。

- a. 関数
- b. オブジェクト
- c. シーケンス
- d. メソッド
- e. プロトコル

Q 8-9 難易度 ★★★

書籍名と著者名のデータ構造を扱える、簡単なクラスを作ってみました。3つある空欄に入るものを選択肢から選んで、「1:a」「2:b」のように答えてください。

```
class Book:
    def 1 (self, title, author):
        self.title = title
        self.author = 2
    def __repr__( 3 ):
        return f"Book('{self.title}', '{self.author}')
```

- a. this
- b. self
- c. author
- d. __init__

Chapter 09

Q 9-1 難易度 ★☆☆

Pythonのモジュールについて、次の選択肢から本書の解説に書いてあることと合致している項目すべてを選んでください。

- a. 実体は.pyファイルだ
- b. 入れものとして存在するだけでプログラムファイルとしては実行されない
- c. 関数とクラスのみ登録でき、変数は登録できない
- d. import文でプログラムに読み込まれる

Q 9-2 難易度 ★☆☆

Pythonのパッケージについて、次の選択肢から本書で解説してあることと異なる説明すべてを選んでください。

- a. 複数のモジュールをまとめるために使われる
- b. 実体は__init__.pyという名前のファイルである
- c. 子供のモジュールはディレクトリの下にまとめられる
- d. Pythonの標準ライブラリにはパッケージは存在しない

Q 9-3 難易度 ★☆☆

次のプログラムで行われていることの説明として正しいものを選択肢の中から選んでください。

```
from datetime import date
```

- a. datetimeクラスのdateメソッドを呼び出している
- b. datetimeからdateクラスをインストールする
- c. datetimeクラスをdateとしてインポートする
- d. datetimeモジュールからdateクラスをインポートする

Q 9-4 難易度 ★☆☆

randomモジュールからchoice関数をインポートしてプログラムで使いたいと思います。「import random」のようにインポートする場合、choice関数を使うのに正しい方法を選択肢から選んで記号で答えてください。なお、「L」は任意のリストであるとします。

- a. choice(L)
- b. random.choice(L)
- c. choice.random(L)
- d. from random choice(L)

Q 9-5 難易度 ★★☆☆

次の文章にある1から3までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

Pythonの 1 と 2 は、プログラムを効率的に作るための重要な要素です。 1 は主に、いろいろなプログラムで共通部品として使えるように関数やクラスを保存して、必要に応じてインポートして呼び出すために利用します。 2 は複数の 1 をディレクトリという形で管理しやすくまとめたものです。 2 にはたいてい 3 という名前のファイルが置いてあって、特別な目的のために利用されます。

- a. ライブラリ
- b. `__init__.py`
- c. モジュール
- d. パッケージ
- e. `__package__.py`

Q 9-6 難易度 ★★☆☆

データの可視化を行うために、AIに書かせたプログラムをコピーして、「import matplotlib.pyplot as plt」としてインポートを行いました。pyplotライブラリのplot()という関数を使いたいのですが、このようなインポートをしたときどのように呼び出すのが正しいでしょうか。選択肢の中から選んでください。引数は省略しています。

- a. `plt.plot()`
- b. `matplotlib.pyplot()`
- c. `pyplot.plot()`
- d. `matplotlib()`

Q 9-7 難易度 ★★★

Pythonのパッケージに含まれる__init__.pyというファイルには特別な役割があります。次の選択肢から本書の解説と合っている項目をすべて選んでください。

- a. パッケージがインポートされるときに実行される
- b. パッケージに必ず含まれている必要がある
- c. 中身を空にしてはならない
- d. このファイルが存在することで、そのディレクトリがパッケージになる

Q 9-8 難易度 ★★★

次のようなtest.pyファイルがあります。別のプログラムで「import test」とし、モジュールとして読み込んだとき起こることとして正しいものを選択肢の中から選んでください。

```
a = 100
print(a)
def foo():
    print(200)
```

- a. 出力セルに「100」と表示される
- b. エラーになる
- c. 出力セルになにも表示されない
- d. 出力セルに「100」「200」と表示される

Chapter 10

Q 10-1 難易度 ★☆☆

Pythonで名前を使って管理されるものを、選択肢からすべて選んで記号で答えてください。

- a. 変数
- b. モジュール
- c. クラス
- d. 組み込み関数

Q 10-2 難易度 ★☆☆

Pythonの名前の説明として、本書の解説と合うものを選択肢の中からすべて選び、記号で答えてください。

- a. 名前はオブジェクトと紐づくように管理されている
- b. 名前は名前空間の上に登録され管理されている
- c. プログラムが終了しても名前は消えない
- d. 関数名を変数名のように扱って代入することはできない

Q 10-3 難易度 ★☆☆

Pythonの名前空間の説明として、本書の解説と合うものを選択肢の中からすべて選び、記号で答えてください。

- a. 組み込み型を除いたすべての名前は名前空間上で管理されている
- b. Pythonではプログラムが終了しても名前空間は残っている
- c. Pythonでは名前空間は1つだけである
- d. 名前空間には辞書型のような仕組みが使われている

Q 10-4 難易度 ★☆☆

Pythonの名前空間には種類があり、上下関係があります。常に存在する組み込み名前空間を一番上位と見たとき、次にくるのはどの名前空間でしょうか。適切なものを選択肢から選んでください。

- a. パッケージ名前空間
- b. モジュール名前空間
- c. ローカル名前空間

Q 10-5 難易度 ★★☆☆

次の文章にある**1**から**3**までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

Pythonでは変数だけでなく関数やクラス、モジュールなどすべてが **1** として実装されています。代入文やdef文、import文のような、名前と **1** を紐づける文がどこで実行されるかによって、名前の所属する名前空間が決まります。たとえば、トップレベルで実行されれば **2** に置かれそうですし、関数内では **3** に置かれます。名前から **1** が参照されるときも、プログラムがどの名前空間に存在するかによって優先される参照先が変わります。Pythonの名前のシステムは、とてもシンプルな仕組みで実装されていることがわかります。

- a. 変数
- b. ローカル名前空間
- c. オブジェクト
- d. モジュール名前空間

Q 10-6 難易度 ★★★

次の文章にある1から3までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

Pythonではオブジェクトやその設計図となるクラスも名前空間のような仕組みを持っています。たとえば、 1 はクラスオブジェクトに紐づいており、これは関数がモジュール名前空間に紐づいているのと関係性がよく似ています。また組み込み型などを代入する 2 だけでなく、 1 自体も 2 で、これは関数名と変数名が同じ名前として操作できる仕組みによく似ています。名前空間上の名前も、 2 も、どちらも 3 のような仕組みで管理されているのです。

- a. インスタンス
- b. アトリビュート
- c. 辞書
- d. メソッド

Q 10-7 難易度 ★★★

次のようなプログラムを実行したとき、出力セルに表示されるものを選択肢から選んで、記号で答えてください。

```
a = 100
def test(a):
    print(a)

test(200)
```

- a. 100だけが表示される
- b. 200だけが表示される
- c. 100と200が1行ずつ表示される
- d. 何も表示されない

Q 10-8 難易度 ★★☆☆

Pythonでは暗黙に置かれる特殊な名前がいくつかあります。このうち、モジュールに置かれる名前で、プログラムの末尾にあるif文の次の空欄のような位置に添えられるものを選択肢の中から選んでください。

```
if _____ == "__main__":  
    # モジュールを直接実行したときに動かしたいプログラム
```

- a. `__file__`
- b. `__name__`
- c. `__all__`
- d. `__init__`

Chapter 11

Q 11-1 難易度 ★☆☆☆

Pythonでブロック内の例外を捕まえるために使われる文を、次の選択肢の中から1つ選んで記号で答えてください。

- a. `try ~ catch`文
- b. `if ~ else`文
- c. `try ~ except`文
- d. `for ~ except`文

Q 11-2 難易度 ★☆☆☆

Pythonの例外について、本書の解説と合うものを選択肢の中からすべて選び、記号で答えてください。

- a. Pythonの例外はエラーだが、文法エラーだけは例外ではない
- b. 例外が発生したときに表示される文字列をストックと言う
- c. 発生した例外が捕捉されないとき、プログラムの実行が止まる
- d. エラーの目的以外で例外が使われることはない

Q 11-3 難易度 ★☆☆

プログラムで人為的に例外を発生させる方法として、正しいものを次の選択肢から1つ選んで記号で答えてください。

- a. `try SyntaxError("文法エラーです")`
- b. `raise SyntaxError("文法エラーです")`
- c. `except SyntaxError as "文法エラーです"`
- d. `new SyntaxError("文法エラーです")`

Q 11-4 難易度 ★☆☆

Pythonのwith文について、本文の内容から判断して間違っているものを選択肢の中からすべて選び、記号で答えてください。

- a. 直後にブロックを伴う
- b. ファイル操作に限定されており、他のコンテキスト管理には使用できない
- c. 「`with open(…) as f:`」のようにすると、ファイルオブジェクトがfに代入される
- d. リソースの確保を行えるが、解放は手動で行う必要がある

Q 11-5 難易度 ★★☆☆

次の文章にある1から3までの空欄にふさわしい選択肢を選んで、「1:a」「2:b」のように答えてください。

Pythonのtry～exceptの後には **1** を置くことができます。直後のブロックは例外が発生する、しないに関わらず必ず実行されるため、tryブロックで確保した **2** の解放を行うプログラムが置かれることがあります。ファイルを閉じる処理のように **2** の確保や解放を伴うプログラムは他にもあり、これを暗黙に実行できるようにしたのが **3** です。

- a. with文
- b. オブジェクト
- c. finally
- d. リソース

Q 11-6 難易度 ★★★

input()関数で入力された文字列を数値型(int型)に変換するプログラムで、数値以外の文字列が入力されたときに警告を表示するようにしたいと思いプログラムを作りました。1と2の2つの空欄に入るものを選択肢から選んで、「1:a」「2:b」のように教えてください。

```
1 :  
    num = int(input("数値を入力してください: "))  
except 2 :  
    print("警告: 数字を入力してください。")
```

- a. as
- b. try
- c. ValueError
- d. raise

Chapter X (総合)

Q X-1 難易度 ★★★

次のプログラムは、リストに日次の歩数を保存し、1週間の平均歩数をround()で丸めた数値を表示する目的で作ったものですが、間違いがあります。1行だけ修正して間違いを直すとして、何行目をどのように修正すればよいでしょうか。なお、正しい答えは「8814」となるはずです。

```
1: steps = [8200, 10100, 7600, 9000, 5000, 12000, 9800]  
2: avg = sum(steps) / 6  
3: print(round(avg))
```

Q X-2

難易度 ★★★

おみくじ用の関数を定義し、占いプログラムを作ろうと思いました。実行してみると、出力セルに「大吉」のような運勢が表示されたあと、「今日の運勢は None」と表示され、期待通りに動きません。「今日の運勢は 大吉」と、メッセージに加えて運勢が表示されるように直したいと思います。1行だけ修正するとしたら、何行目をどのように直せばよいでしょうか。

```
1: import random
2:
3: def omikuji():
4:     items = ["大吉", "中吉", "小吉", "凶"]
5:     print(random.choice(items))
6:
7: result = omikuji()
8: print(f"今日の運勢は {result}")
```

有名な戯曲の一編を文字列に代入し、単語に分割して出現頻度を数えるプログラムを作りたいと思います。その際、「,」(コンマ)のような記号を取り除き、区切った単語は小文字にし、「the」「and」「is」のような単語は除去する前処理を行ってから、単語を数えたいと思います。プログラムを動かしてみると、期待通りの結果になりません。1行だけ修正するとしたら、どの行をどのように直せばよいでしょうか。正しい出力は「to」が3個、「be」が2個含まれる結果になるはずです。

```
01: from collections import Counter
02: from string import punctuation
03:
04: # シェイクスピアのハムレットより
05: text = """To be, or not to be, that is the question:
06: Whether 'tis nobler in the mind to suffer
07: The slings and arrows of outrageous fortune,"""
08:
09: # ,のような記号を取り除く
10: norm = "".join(ch for ch in text if ch not in punctuation)
11: norm = norm.lower() # 小文字に
12: words = norm.split() # 単語を区切りリストに変換
13: words = [word for word in words if word in {"the", "and",
14: "is"}]
14: cnt = Counter(words)
15: print(cnt.most_common())
```

解答と解説

Chapter02

Q 2-1 の解答

答え：b

本書では、変数には代入されたデータの内容がわかる名前をつけることを推奨しています。その文脈から考えると、「price」が適切です。「x」や「a」では抽象的で内容がわかりません。「tax_rate」は「税率」という意味なので、商品の値段としてはふさわしくありません。

Q 2-2 の解答

答え：b

book_price（書籍の価格）とあるので、選択肢の中では **b** がもっとも適切です。

Q 2-3 の解答

答え：c、d

Python の計算式で優先順位をつけるために使えるのは丸括弧（`()`）のみです。計算に波括弧は使えないため、**c** は間違いです。また、累乗を計算するために使う演算子は「`**`」ですので、**d** も間違いです。

Q 2-4 の解答

答え：b、d

文字列のリテラルを問う問題です。**a** は日本語（カタカナ）に見えますが、クォーテーションで囲まれていないため Python はこれを変数名として扱います。**b** と **d** は英字と数字ですが、ダブルクォーテーションで囲まれているので文字列リテラルとして扱われます。**c** は数値リテラルです。

Q 2-5 の解答

答え：a、c、d

演算子を使った計算ではデータの種を合わせるのが基本です。**a** は数値と数値の足し算なのでエラーになりません。**b** は数値と文字列の足し算で、種類が異なるためエラーになります。**c** は文字列の足し算でエラーにならず、2つの文字列を連結して「1020」という計算結果になります。**d** は文字と数値の掛け算で種類が異なるのですがエラーにならず、「0」という文字列を100個並べる、という計算結果になります。

Q 2-6 の解答

答え：aと4、bと3、cと2、dと1

比較演算子の機能についての確認問題です。**a** は左右が等しい（一致する）、**d** は左右が等しくない（一致しない）、**b** は左側が右側より大きい、**c** は左側が右側と等しいか大きい（以上）となります。イコールと不等号の組み合わせは「等しいか大きい・小さい」とすると覚えやすいです。イコールを含まない不等号は「等しい場合」を含まないわけです。

Q 2-7 の解答

答え：a、c

Python で条件分岐を行う if 文の機能を問う問題です。「ゆっくり休みましょう」と出力されるブロックが実行されるには、「temperature >= 37」の条件が成り立つ数値が変数に代入されている必要があります。そう考えると **a** と **c** になります。余談ですが100度の体温というのはあまり考えたくないですね。

Q 2-8 の解答

答え：1:b、2:c、3:a

変数は、Python のプログラムでデータの入れものとして使われます。中にどんな種類のデータが入っているのか、読むだけでわかる変数名をつけることを本書では強くお勧めしています。x や a のような抽象的で短い変数名はお勧めしていません。

Q 2-9 の解答

答え：1:a、2:d、3:b

関数を作り、使う方法に関わる用語を問う問題です。関数定義では、入力となる変数を括弧の中に記述しますが、これを「引数」といいます。関数を使うことを「呼び出す」と言いますが、呼び出すとき括弧の中に入っているデータが引数に代入され、ブロックの中で使われます。関数の出力となるデータのことを特別に「戻り値」と呼びます。どれもプログラミングで一般的に使われる用語なので、ぜひ覚えてください。

Q 2-10 の解答

答え：a、d、e

組み込み関数の働きを問う問題です。**a** は数字の文字列を数値に変換する関数なので 100 になります。**b** は逆に数値を文字列にするので "100" という文字列になります。**c** は 0 と 100 のうち小さい方が戻り値になるため 0 になります。**d** は 100 の 1 乗を返すので 100 になり、**e** は 100 と 0 のうち大きい方を返すので 100 になります。

Q 2-11 の解答

答え：b

基本パターンに沿って見ると、まず入力となる **a**、または **c** のプログラムがきます（順序不同）。**b** は処理（計算）を行っているので、3 つの中では最後にきます。さらにこの後に、「print(area)」のようなプログラムが続くことが予想されますね。

Q 2-12 の解答

答え：1:d、2:b、3:a

関数を定義するプログラムをどう書くかについて問う問題です。関数定義は「def」からはじめます。関数の中では、引数の base（底辺）と height（高さ）を掛け算して、結果を変数に代入しています。空欄 **2** には結果を代入する変数として area（面積）が入ります。最後に空欄 **3** に「return」を書くことで、結果（area）を関数の戻り値とします。

Chapter03

Q 3-1 の解答

答え：c

リストのインデックスは0からはじまります。「1 から数えて 3 番目」の要素を取り出したいときは、3 から 1 を引いた「2」をインデックスとして与えることになり、c が正解です。

Q 3-2 の解答

答え：1: c、2: d、3: b

文字として読んでもなんとなくわかるかもしれませんが、辞書が横長に書かれているので、まずキーと値を読み分けます。

1 は name (名前) や address (住所) というキーからアドレス帳アプリの項目 (c) がふさわしいとわかります。2 のキーは漢字ですが、種や類などから生物の分類 (d) となります。3 のキーになっている英単語は少し難しいかもしれませんが、temperature (気温) や humidity (湿度) と調べてわかると気象の情報 (b) が適切となります。

Q 3-3 の解答

答え：b

辞書の要素にアクセスする方法を問う問題です。この場合、角括弧内に文字列としてキーを与えることで、要素を取り出すことができます。a は数値なので間違いです。b は「霊長類」という値のキーになっている文字列を与えているので正解です。c はドットで区切ってキーを与えているので間違い。d は引用符で囲まない「類」を与えているため間違いです。

Q 3-4 の解答

答え：b

Chapter03 で取り上げたものでいうと、a はリスト、辞書、タプルすべてについて成り立ちます。リストと辞書は要素の置き換えができるため、b はタプルのみに成り立ちます。c はリストとタプルについて成り立ち、d は辞書についてのみ成り立ちます。

Q 3-5 の解答

答え : 1:d、2:b、3: e

関数の `range()` がどのような数を産み出すのかを問う問題です。リストなどのインデックスと同じく、引数が 1 つのときは 0 から始まり引数 - 1 までの数を産み出すのがまず押さえて欲しいポイントです。引数が 2 つあるときは、第 1 引数が開始、第 2 引数 - 1 までの数を産み出します。

1 は **d** で、**f** と間違えて欲しくないところです。2 は **b** で **c** と間違えないでください。3 は少し意地悪な問題で、「2 から 3 - 1 まで」なので **e** となります。

Q 3-6 の解答

答え : 1:c、2:d、3:a

Chapter03 で解説した組み込み型の特徴を問う問題です。リストは同じ種類のデータを並べるのに使われるのに対し、辞書とタプルは異なる種類のデータを並べるのに使われます。リストとタプルはよく似ているのですが、初心者の方は主にリストを使うことが多いはずです。

Q 3-7 の解答

答え : a、c、d

a は辞書に存在しないキーに対して代入をしようとしています、要素の追加になるのでエラーにはなりません。**b** はキーを使って要素を参照した結果を変数 (`temp`) に代入しようとしています、"`temp`" というキーの要素は辞書に存在しないのでエラー (`KeyError`) になります。**c** は既存のキーに対する代入でエラーにならず、要素の入れ替えが起こります。**d** は既存のキーを使った参照になり、エラーにはなりません。

Q 3-8 の解答

答え : b、d

リストに角括弧を添えてインデックスを使い要素にアクセスする方法を問う問題です。リストのインデックスは 0 から始まりますので、最後の要素を指すインデックスは「3」になります。**b** と **d** は 0 から 3 までのインデックスを指定しているのでエラーにはなりません。**a** は 3 を超える数値をインデックスに添えているためエラーになります。**c** は関数の戻り値を角括弧の中に添えています、`len(points)` の戻り値は要素数の「4」なのでエラーになります。ちなみに、インデックスに「-1」を与えると、リストの要素数に関わらず最後の要素を指し示すことができます。

Q 3-9 の解答

答え：b

human という変数に入っている辞書がループの in に添えられています。このとき、キーが順番にくり返し変数 (category) に入り、ループが実行されます。選択肢のうち **b** と **c** が辞書のキーですが、**c** はプログラムにある辞書と順番が逆転しているため、**b** が正解です。

Q 3-10 の解答

答え：b、e

抽象的な数値のリストを使った計算を頭の中で行う問題です。リストのインデックスを数値に置き換えて計算をするのが 1 つの解き方です。**a** は「 $(3 + 10) \times 3 = 39$ 」なので×です。**b** はリストの最後の要素を指しているので○です。**c** は「 $10 \div 2 \times 3 = 15.0$ 」なので×になります。**d** は要素を超えるインデックスが与えられているのでエラーになります。**e** は「 $2 \times (2 + 3) = 10$ 」なので○ですね。実際に自分でプログラムを入力してみても確認できます。

Q 3-11 の解答

答え：1:e、2:c、3:d

空欄 1 と空欄 3 には、プログラムの他の部分に書かれている変数を埋めます。空欄 2 は for 文の文法についての知識を問う問題です。選択肢のうち **a** は関数定義のための文法なので除外し、**b** はプログラムには出てきませんので除外します。**c**、**d**、**e** を使って空欄を埋めればよいわけです。空欄 1 は、コメントにあるように「合計を足していく変数」として選択肢 **e** の total を選ぶのが正解です。空欄 2 は、for 文に必ず添える選択肢 **c** の in を選びます。if 文のブロックでは、合計 (total) にリストの要素を足していくプログラムを書きます。リストの要素は 1 つずつくり返し変数 (point) に代入されていきます。total に足している空欄 3 は、くり返し変数の選択肢 **d** の point となります。

Q 3-12 の解答

答え：b

プログラムの 1 行目は関数をインポートしています。2 行目は random() 関数を呼び出し、結果に 6 を掛けた数値を変数に代入していて、3 行目で出力セルに表示しています。random() が返す数値は、0 を含み 1 より小さい数値です。**d** は random() の返す数値そのもののので除外されます。**a** は「1 以上」となっているところが関数の解説と違うので除外。1 未満のランダムな数値に 6 を掛けると 0 以上 6 未満になるので **b** が正解です。

Chapter04

Q 4-1 の解答

答え：b、c

a は数字で書かれていますが引用符で囲まれているので文字列になります。**d** は数字だけで構成されていますがリストです。**b** は 16 進数の数値リテラル、**c** は見た目で数値とわかりますね。

Q 4-2 の解答

答え：b、d

a は三重引用符を使った文字列リテラルです。**b** はよく見ると引用符の対応が間違っているためエラーになり文字列ではありません。**c** はエスケープシーケンスを含んだ文字列、**d** は引用符で囲まれていないため変数名として扱われます。**e** は数字で構成されていますが引用符で囲まれた文字列です。

Q 4-3 の解答

答え：b、d

a は角括弧で囲まれているのでリストのリテラルです。**b** は丸括弧で囲まれているためタプルです。**c** は文字列なのでタプルではありません。**d** は文字列がコンマで区切られていて括弧がありませんが、タプルとして扱われる特別な記法です。

Q 4-4 の解答

答え：b、d

リスト型のメソッドのうち、リストの要素を書き換える破壊的操作を行うメソッドは引数を返しません。4 つのメソッドのうち、`count()` は引数として渡された要素の数をカウントするメソッドのため引数を返します。`remove()` は引数の要素を取り除く破壊的メソッドのため引数を返しません。`index()` は引数の要素があるインデックスを調べて返すメソッドなので引数を返し、`clear()` は要素をすべて消去するため引数を返しません。結果として **b** と **d** が答えとなります。

Q 4-5 の解答

答え : 1:d、2:b、3:c

言葉の定義の問題です。データの種類のことを「型」と呼び、Python ではプログラムを作るために使う基本的な型を特別に「組み込み型」と呼んでいます。「リテラル」は Chapter03 で簡単に解説し、Chapter04 ではたびたび登場する言葉です。難しい言葉ですがぜひ覚えて欲しいので、あえてここで設問に加えました。

Q 4-6 の解答

答え : 1:c、2:d、3:b

オブジェクト指向の概略を解説した文章に、正しい言葉を埋める問題です。「オブジェクト」が操作の主体（主語）となり、動作を指定する部分は「メソッド」と呼ばれています。オブジェクト指向はオブジェクトを中心にすえてプログラムを作る手法につけられた名前です。ちなみに選択肢にある「アトリビュート」は Chapter07 以降で解説する言葉で、オブジェクトが持つデータのことを指します。

Q 4-7 の解答

答え : b、c

a は `in` 演算子を使って文字列の中に特定の文字列が含まれるかどうかを判定しています。in の右側に左側の文字列がないため False になります。**b** は文字列の大小を比較する条件式でプログラムとしては意味がありませんが True になります。**c** は「`==`」演算子を使って文字列が同じかどうかを比較しているため True、**d** は **b** の文字列を `int()` 関数を使って文字列を数値に変換してから比較しており、数値の大小が評価され False になります。

Q 4-8 の解答

答え : 1:a、2:d、3:b、4:c

スライスに関する理解を確かめる問題です。**1** は 0 から 1 (2 の 1 つ前) までを取り出すスライスなので **a** になります。**2** はリストにインデックスを添えて要素を取り出しているため **d** が正解です。**3** も 0 番目の要素だけを取り出すのですが、スライスを使っているのでリストになるため **b** です。**4** はスライスの足し算をしてリスト全体になるため **c** が正解です。

Q 4-9 の解答

答え：a、d

a はリストのスライスと代入を使って複数要素を入れ替えるプログラムで、エラーになりません。**b** はタプルの要素を入れ替えようとしているためエラーになります。**c** はタプルに存在しない `clear()` という破壊的メソッドを実行しようとしているためエラーになります。**d** の `count()` メソッドはタプルとリストの両方に存在するメソッドで、エラーになりません。

Q 4-10 の解答

答え：a

メソッドを使って降順にソートすることを考えます。**b** はリストを変数に代入する行なので最初にきます。**c** で昇順（小さい順）でソートをした後、**a** の `reverse()` で順番を入れ替え、降順にすることで目的のプログラムになるため、**a** が最後にきます。なお、「`lons.sort(reverse=True)`」のようにしても降順ソートが行えます。

Chapter 05

Q 5-1 の解答

答え：a、c、d

辞書のキーについての理解度を確認する問題です。キーとしては変更不可能な組み込み型のみが使えるため **a** と **d** は正しいです。**b** は本書の解説では「重複できない」とは書いてないのですが、既存のキーを指定して代入すると上書きになることから、キーは重複できないことになるため間違いになります。また、キーのみをシーケンスとして取り出すとき、後に追加したもののほど後に現れ、登録順が保存されるため **c** は正しいです。

Q 5-2 の解答

答え：c

`if` 文で辞書のキーの検査をどのようにすればよいのかを問う問題です。メソッドを組み合わせることもできますが、ここでは比較演算子の `in` を使う選択肢に正解があります。**b** と **c** の違いは調べたいキー（4）が左にあるか右にあるかですが、右にある **c** が正解となります。

Q 5-3 の解答

答え：a、c、d

シーケンス (sequence) とは、「連続した」という意味の英単語からきた言葉で、リスト、タプル、文字列のような Python の組み込み型の一部が持つ性質に対する分類名として使われています。連続した要素を持つことから想像できるように、シーケンス型の仲間は複数の要素を持ち、順番があり (b は間違い)、for 文の in の右側に添えることができます。角括弧にインデックスを添えることで要素にアクセスできるのも特徴です。また要素を重複して持つことができます。

Q 5-4 の解答

答え：b

設問のイミュータブル (immutable) は変更が不可能な組み込み型に使われる分類名です。変更が不可能なので a は間違いです。イミュータブルな型は辞書のキー、set 型の要素になることができるため、b は正しく、c は間違いです。「変更ができない」のは「破壊的操作ができない」と同じ意味なので、d は間違いです。正しいのは b だけになります。なお、ミュータブル (mutable) は変更が可能な組み込み型を指します。

Q 5-5 の解答

答え：1:d、2:b、3:c

空欄 1 と空欄 2 に、キーと値のどちらかが入ることは文脈からわかるかも知れません。キーにできる組み込み型は限られていることから、空欄 1 がキーとなります。また、組み込み型には変更できるものとできないものがあり、このうちキーにできるのは変更できないものだけです。

Q 5-6 の解答

答え：1:c、2:b、3:a

Python 自体というよりその周辺にあるドメイン知識を問う性質の問題で、ちょっと難しかったかもしれません。文字列型はエンコード済みのデータ、bytes 型はエンコード前のデータを扱います。本書の解説では前者を「飼い文字列」、後者を「野良文字列」と呼んでいました。ちなみに、ユニコードはエンコード方式につけられた名前です。

Q 5-7 の解答

答え：b、c、d

どれも辞書に「存在しないキー」を使った操作です。**a** は辞書に存在しないキーを使って要素を参照しようとしているのでエラーになります。**b** は `get()` メソッドに存在しないキーを与えて要素を取り出そうとしていますが、第 2 引数を与えられているため "IV" という文字列が返ってきてエラーになりません。**c** は `in` 演算子を使った要素の検査で `False` となりエラーになりません。**d** は存在しないキーに対して代入を行っていて、要素が追加されエラーになりません。

Q 5-8 の解答

答え：1:d、2:a、3:b

`set` 型の性質に関する解説の穴埋め問題です。空欄 1 は「追加」か「変更」か迷うところですが、空欄 3 が「変更」なので、空欄 1 は「追加」になります。`set` 型は「集合型」とも呼ばれ、これが空欄 2 の選択肢にあたります。

Q 5-9 の解答

答え：1:a、2:c、3:d

空欄 1 は `=` (イコール) の左にあることから変数名が入り、プログラムの他の部分から **a** とわかります。空欄 2 は辞書型のメソッド名なので **c** になります。空欄 3 は文字列の自身で厳密には **d**、**e** どちらでもいいのですが、`get()` メソッドの機能を考えると、辞書のキーが存在しなかったときに返される文字列が第 2 引数にくるので、**d** が適切です。

Q 5-10 の解答

答え：a → c → b

本書の解説では「ファイル処理にも、プログラムの基本形のような手順がある」と解説した部分の確認問題です。**a** が入力、**b** が処理と出力に相当します。**c** は呼ばなくてもよい場合もあります。ただ、いったん閉じたファイルに対して **b** の操作をしようとするとエラーになります。

Chapter06

Q 6-1 の解答

答え：b、d

本書で解説した式と文は、少し抽象的でわかりづらかったかも知れません。でも、ここを理解すると Python の理解度がグッと上昇しますのでぜひ押さえてください。式は、プログラムの中で値を持った要素（b）で、計算式や戻り値を、値として持つ関数呼び出しが相当します（d）。Python の文は、代入文や if 文のように必ず改行を伴いますが、式は改行を伴いません。

Q 6-2 の解答

答え：b

Python の比較演算子「==」では、値が正しくかつ型も等しいときにはじめて True となります。a はタプルとリストの比較、c は文字列と bytes 型の比較、d は数値と数字だけで構成された文字列の比較で型が異なるためどれも False になり、b だけが True になります。

Q 6-3 の解答

答え：a、c

Python では、「空でない式」は bool 型（真偽値）として扱うと True となり、このテクニックを使ってプログラムを短く書くことがよくあります。a は空でない文字列なので True、b は空のリストなので False です。c は空でない set 型で True、d は少しトリッキーなのですが、リストに対してスライスを適用して要素 0 のリスト（空のリスト）を取り出しているため False になります。

Q 6-4 の解答

答え：1:c、2:a、3:b

ループブロックの中に置いて使う文と、節と呼ばれる機能についての問題です。break 文があると、ループは途中で終了します。continue 文があると、その後のブロックは実行されず、（あるときは）次の要素を使ってループブロックが実行されます。if 文でも使われる else 節は、ループが一度も break されずに正常終了したときだけ直下のブロックを追加で実行します。

Q 6-5 の解答

答え：1:d、2:a、3:c

代入文について式と文を通して再定義した、本書の解説を確認するための問題です。空欄 1 と空欄 2 には「式」、または「文」のどちらかが入ることは類推できると思います。代入は文なので空欄 2 が「文」になり、空欄 1 は「式」となります。左側にタプルを置いた代入を「アンパック代入」と呼ぶことから、空欄 3 は c になります。残った選択肢の複合演算子は「+=」のように演算と代入を同時に行う演算子のことを指すため、ここでは選ばれません。

Q 6-6 の解答

答え：1:c、2:d、3:b

関数内で作られるローカル変数と、関数外のグローバル変数について、本書で解説した内容を確認する問題です。空欄 2 と空欄 3 のどちらかが「グローバル」「ローカル」になります。「関数内でしか生存できない」「関数内部からしか見ることができない」とある空欄 2 が「ローカル変数」です。「関数の外部と内部どちらからも見ることができる」のは「グローバル変数」になります。空欄 1 は変数を作る方法について解説している部分です。Python では「代入」によって変数を定義することをぜひ覚えておいてください。

Q 6-7 の解答

答え：1:d、2:b、3:c

空欄 1 は「def 文の直下に」とあることから「ドックストリング」とわかります。空欄 2 は「文字列でもよい」と書いてあるので「アノテーション」がより適切です。アノテーションは文法的には式なので、文字列、呼び出し可能オブジェクト、また変数やリスト内包表記などいろいろなものを置くことができます。引数や戻り値の型を柔軟に指定する方法としていろいろなテクニックが存在します。

Q 6-8 の解答

答え：c

引数名の前にアスタリスク (*) がつく「可変長の位置引数」を持つ関数の機能について確認する問題です。受け取った引数はリストになりますが、関数呼び出しをするときは引数をコンマで区切って渡すため、a の説明は消えます。引数をいくつ受け取ることができるかは関数内部のプログラムに依存します。args が空の場合は、for ループが一度も実行されず 0 が返ってきます。つまり、引数はゼロ個でもよいことになり、c が正解です。

Q 6-9 の解答

答え : 1:c、2:a、3:b

リスト内包表記は for 文のキーワードをうまく使って作られています。for 文のプログラムは縦に長くなりますが、リスト内包表記を使ったプログラムは横に長くなることに気をつけて、また変数の位置が入れ替わることを念頭に置いて、空欄を埋めていきます。空欄 1 は結果を代入する変数名なので **c** です。空欄 2 はたいていくり返し変数が置かれるので **a** になり、ここでは int() 関数の引数になっています。空欄 3 はシーケンスが添えられる場所なので **b** が正解です。

Q 6-10 の解答

答え : c

リスト内包表記の前半までは Q6-9 と同じで、str_list の文字列を int 型に変換したリストを作るリスト内包表記であることがわかります。この時点で **b** が消えます。違うのはその後の if です。この条件を見ると「" not in num_str」となっており、くり返し変数の文字列に「」（ドット）がない場合のみ、リストの要素として評価することがわかります。結果から "200.0" という要素が除外されることになるので、**c** が正解です。

Q 6-11 の解答

答え : d

デフォルト引数を持つ関数に関する問題です。第 2 引数を与えない状態では、tax_rate に 0.2 が代入された状態、つまり 20% の税金を追加した値が返ってきます。200 だけ引数に与える **a** は 240 (200×1.2) となり、**b** の 180 だけでは 216 (180×1.2) になります。**c** は tax_rate が 10 なので 200×11 で 2200 です。**d** は tax_rate が 0 なので 200×1 で 200 になり、**d** が正解です。

Q 6-12 の解答

答え : 1:a、2:d、3:c

選択肢の位置から、どの選択肢も変数名である、と気づくところが出発点になります。**b** の range は関数名なのでこの時点で選択肢から外れます。空欄 1 はリストとなる変数名で、enumerate() の引数にも渡されている **a** となります。また enumerate() が順番、要素を次々に返すことを思い出すと、空欄 2 は print() の引数になっている f 文字列のうち num が、空欄 3 は name が入ります。

Chapter07

Q 7-1 の解答

答え：a、c、d

本書の解説では、まずオブジェクトは「データ構造の一種」とであると説明しました。また、オブジェクトは辞書や数値のような組み込み型と同じように「変数に代入」できるので **b** は間違いです。オブジェクトは「アトリビュート」を持っていて、データ構造の実体はここに収められます。また、オブジェクトを作るときは、クラス名を関数のように呼び出して作ります。

Q 7-2 の解答

答え：b、c

オブジェクト名からドット (.) を挟んで記述する変数のようなものである、というのがアトリビュートの基本です。変数と同じように振る舞うため、**a** と **d** は正しい説明です。区切る記号が間違っているため **b** は間違いです。また何度でも違う値を代入できるため **c** も間違いです。

Q 7-3 の解答

答え：c

Python でクラスを定義するためには `class` 文を使います。`def` 文は関数定義に使われ、`for` 文はループに使われます。なお、Python には `object` 文というのはありません。`object` はすべてのオブジェクトの親となる基底クラスで、すべてのクラスは `object` を暗黙に継承しています。

Q 7-4 の解答

答え：a

Python のメソッドは、`def` 文を使って定義します。関数と違い、クラス定義のインデントブロック内で定義されたものがメソッドとなります。

Q 7-5 の解答

答え：1:c、2:a

`__init__` はオブジェクトを作るときに最初に呼ばれる「初期化メソッド」の名前です。`self` はメソッドの「第 1 引数」として置かれ、オブジェクトそのものが暗黙に代入されます。メソッド内部では、`self` を経由してアトリビュートやクラスのメソッドにアクセスできます。

Q 7-6 の解答

答え：2100

`datetime` クラスは日付と時刻をデータ構造として持つクラスです。`a_day.year` のアトリビュート部分は名前から西暦であることがわかり、2 行目の最初の引数として 2100 という西暦が渡されていることから、2100 が出力されます。

Q 7-7 の解答

答え：1:d、2:a、3:c

本書で「オブジェクトはデータ構造である」と解説した内容の確認問題です。「変数の集まりや辞書で扱えることができる」とあるため、空欄 1 は「データ構造」とわかります。初出の空欄 2 はクラスでもよさそうなのですが、直後に「指向」と続くことからオブジェクトがふさわしいとわかります。空欄 3 は「オブジェクトに紐づいてデータを代入する」とあることから、アトリビュートとなります。

Q 7-8 の解答

答え：1:c、2:b、3:d

Python のクラスの作り方と、オブジェクトの作り方についての確認問題です。空欄 1 は「class 文から続く」とあるのでクラスとわかります。初期化に使われる `__init__` を指していることから空欄 2 はメソッドです。空欄 3 はデータ構造でもよさそうですが、「代入する」とあるのでアトリビュートがより適切です。

Q 7-9 の解答

答え : b

Python でクラスからオブジェクトを作るには、クラス名を関数のように使って呼び出します。初期化メソッド (`__init__`) を見ると引数が 2 つあり、それぞれ `name` (商品名) と `price` (価格) を代入するようになっているため、**b** が正解になります。**a** はクラス名を関数として呼び出していますが、引数がないため間違いです。**c** はリストのリテラルでエラーにはなりませんが、`Product` クラスのオブジェクトは作成されません。

Q 7-10 の解答

答え : 1:e、2:b、3:c

クラス定義の文法を確認する穴埋め問題です。Python のクラス定義は `class` 文から始めるので、空欄 **1** は `class` になります。空欄 **2** はクラス定義のメソッド定義で、`def` がきます。空欄 **3** は面積を計算している部分ですが、`Rectangle` オブジェクトが持つ 2 つの属性のうち `width` がきます。

Chapter 08

Q 8-1 の解答

答え : a、d

Python の組み込み型と特殊メソッドに関する知識を確認するための問題です。組み込み型はすべてメソッドを持っているため **a** は間違いです。また、角括弧を使った要素のアクセスのように、メソッドを使っていないように見える操作でも、内部ではメソッドが呼び出されているため **b** は正しいです。同じ演算子の処理には同じ名前のメソッドが割り当てられているため **c** も正しいです。**d** は少し難しいかもしれませんが、任意のオブジェクトを文字列型に変換するとき、前者が持っている特殊メソッドが呼び出されるため間違っています。

Q 8-2の解答

答え：c、d

Python の特殊メソッドに関する知識の確認問題です。特殊メソッドは、2つのアンダースコア（`__`）で囲まれた英小文字のメソッド名がつけられていますので、**a** と **b** は間違いです。演算やインデックス、キーを使った要素のアクセスを実際に行っているため、暗黙に呼ばれます。また二重のアンダースコア（ダブル・アンダースコア）で囲まれることから「ダンダーメソッド」とも呼ばれています。そのため、**c** と **d** は説明として正しいことになります。

Q 8-3の解答

答え：d

Python で値を暗黙に文字列に変換するときの形式を「repr」と言います。「印字可能な表現を含む文字列」（a printable representation of an object）の略で、「プログラムとして入力したときに元のオブジェクトに戻る文字列」という意味だと、本書で解説したの思い出してもらえれば正解できるはずです。

Q 8-4の解答

答え：1:b、2:c、3:a

`__str__()` はオブジェクトが `str()` 関数の引数として渡されたときに暗黙に呼ばれ、オブジェクトを文字列に変換して返すので **b** です。`__len__()` はオブジェクトの要素数（長さ）を返し、`len()` 関数の引数として渡されたときに暗黙に呼ばれるので **c** です。`__hash__()` はオブジェクトのハッシュ値を計算する関数ですから **a** になります。`hash()` に引数として渡されたときに暗黙に呼ばれ、辞書のキーや `set` 型の要素になることができるオブジェクトはこの特殊メソッドを持っています。

Q 8-5の解答

答え：b

親となるクラスの機能を引き継ぎながら、必要な部分だけを上書きして拡張するのは「継承」です。

Q 8-6 の解答

答え：c

Python でクラスを継承するときの文法の確認問題です。クラスを継承するときは、class 文に続き子クラスのクラス名を書き、括弧の中に親クラス名を書きます。c が正解です。

Q 8-7 の解答

答え：d

クラスを継承するとき、親クラスにあるのと同名のメソッドを子クラスに定義することを「オーバーライド」といいます。Python でオーバーライドされたメソッドは上書きになり、子クラスのメソッドだけが呼び出され、親クラスのメソッドは呼び出されません。特殊メソッドであっても通常のメソッドであっても同様です。

Q 8-8 の解答

答え：1:d、2:b、3:e

Python のオブジェクト指向実装に関する発展的な知識を確認する問題です。空欄 2 は最後に「指向」を伴っているので「オブジェクト」とわかりやすいかもしれません。空欄 1 は「操作」とあり、また「__add__」という特殊メソッドから「メソッド」です。空欄 3 は消去法で「プロトコル」を選べると思います。本書ではあまり出てこない上に、少し難しい用語ですが、ぜひ覚えて欲しいのであえて出題しました。

Q 8-9 の解答

答え：1:d、2:c、3:b

空欄 1 は初期化メソッドなので __init__ になります。空欄 2 は初期化メソッドで受け取った引数をアトリビュートに代入している行なので第 3 引数にある author です。空欄 3 はメソッドの第 1 引数 self になります。「__repr__」という名前のメソッドは repr を表示するためのメソッドで、戻り値として Book クラスを初期化できるような文字列を返しています。

Chapter09

Q 9-1 の解答

答え：a、d

Python のモジュールの実体は「.py ファイル」で、import 文で読み込まれるときにはプログラムファイルとして実行されます。そのため **a** と **d** が正解です。**b** は「プログラムとして実行されない」と書いてあるため間違いです。また、モジュールには関数やクラスだけでなく、定数のような変数も登録できるため、**c** も間違いです。

Q 9-2 の解答

答え：b、d

Python のパッケージは複数のモジュールをまとめるために使われるので、**a** は正しいことを言っています。またパッケージの実体はディレクトリで、__init__.py ファイルは必ずしも必要ないことから **b** の説明は本書の解説と異なります。**c** は正しいことを言っています。標準ライブラリにはモジュールを機能ごとにまとめるためにたくさんのパッケージが存在するため **d** は間違っています。

Q 9-3 の解答

答え：d

Python でインポートを行うときの常套句「from ~ import」についての理解を確認する問題です。**a** と **b** は「呼び出す」「インストールする」という文言から間違いとわかります。**c** は「として」とあるのが間違い。**d** が正解となります。

Q 9-4 の解答

答え：b

from を使わずモジュールそのものをインポートしたとき、その子供として登録されている関数を呼び出す方法を問う問題です。「from random import choice」なら **a** でよいのですが、「import random」のように random モジュールをインポートしているので **b** が正解です。**c** はライブラリ名と関数名が逆順になっているため間違いになり、**d** はかなり混乱している感じがしますが、文法エラーになります。

Q 9-5 の解答

答え：1:c、2:d、3:b

Python のモジュールとパッケージの構造を理解しているかどうかを確認する問題です。空欄 1 はライブラリでも意味は通りますが、「モジュール」を選んでほしいところです。空欄 2 はモジュールをまとめるとあるので「パッケージ」、空欄 3 は「__init__.py」です。

Q 9-6 の解答

答え：a

「as」を使って別名でインポートしたとき、どのようなプログラムを書けばよいのか、という知識を確認する問題です。「pyplot ライブラリの plot()」を使いたいけど、as によって pyplot が plt に置き換わっていることがわかれば、a が選べるはずです。b は「import matplotlib」としたなら正しく、c は as を使わず「from matplotlib import pyplot」とインポートしていたなら正しいことになります。d は使いたい関数が出てこないで明確に間違いです。

Q 9-7 の解答

答え：a

パッケージをインポートするとき、特別な処理を行うために利用されるのが「__init__.py」です。処理を行うときには、__init__.py が暗黙に実行されるので a は正しいです。ただし、存在しなくてもよく、空にしておいてもよいので b、c、d は間違いになります。

Q 9-8 の解答

答え：a

モジュールをインポートしたときには、.py ファイルとしてプログラムが実行されることを確認する問題です。2 行目の print() 関数が実行され、出力セルに変数 a の値が出力されます。その下の def 文も実行されますが、関数が定義されるだけなので出力セルにはなにも表示されません。a が正解となります。

Chapter10

Q 10-1 の解答

答え：a、b、c、d

この問題の選択肢のものは、Python ではすべて「名前」として管理されています。他にも、関数やパッケージなど、「～名」と呼ばれるものはすべて名前に紐づいて管理されています。

Q 10-2 の解答

答え：a、b

名前には必ず実体となるオブジェクトが紐づいていて、名前を管理するための仕組みを「名前空間」といいます。**a** と **b** は正しいことになります。名前はプログラムが終了すると消えてしまうため **c** は間違いです。また Python では関数であっても代入によって名前に紐づくオブジェクトを置き換えることができるので、**d** も間違いです。

Q 10-3 の解答

答え：d

Python の名前空間に関する理解を確かめる問題です。Python では組み込み型を含めたすべての名前が名前空間上で管理されているため **a** は間違いです。また、プログラムが終了すると名前空間は保存されないため **b** も間違いです。Python の名前空間は 3 種類あるため **c** も間違いになります。**d** は名前空間の仕組みの解説として本書でも同様の解説をしています。

Q 10-4

答え：b

本書の解説では、名前空間の順位を、組み込み、モジュール、ローカルの順で上下関係があるように図示して解説しました。「組み込み」の下にくるのは「モジュール名前空間」です。ちなみにパッケージ名前空間というものはありません。

Q 10-5の解答

答え：1:c、2:d、3:b

Chapter10 だけでなく、12-03 節でも解説したことを確認する問題です。Python のオブジェクト指向は名前とオブジェクトを紐づけることで成り立っていますので、空欄 1 は「オブジェクト」となります。空欄 2 と空欄 3 は、共に名前空間を選ぶ選択肢で、空欄 2 はトップレベルとあることから「モジュール名前空間」、空欄 3 は関数内とあることから「ローカル名前空間」とわかります。

Q 10-6の解答

答え：1:d、2:b、3:c

Python のクラスシステムについて名前空間と対比して確認する問題です。空欄 1 は関数と似ているとあることから「メソッド」とわかります。空欄 2 は「組み込み型などを代入する」とあるので、「アトリビュート」と類推できます。空欄 3 は本書でたびたび解説していたので答えて欲しいところですが、インスタンスは入らなそうなことから「辞書」となります。

Q 10-7の解答

答え：b

モジュール名前空間上の変数とローカル変数の挙動に関する問題です。まず test() 関数では、引数「a」の値を表示しています。モジュール名前空間上で「a」という同名の変数が定義されていますが、関数の中では上書きされます。最後の行で test() 関数を呼び出しているとき、引数として「200」が与えられているので、この値の「200」が表示されます。

Q 10-8の解答

答え：b

プログラム実行時、モジュールに対して暗黙につけられる名前のうち、モジュールがどのように実行されたかを示す情報として文字列が代入される名前を問う問題で、答えは「__name__」です。ネットで公開されているプログラムでもよく見かけるので、ぜひ覚えてください。

Chapter11

Q 11-1 の解答

答え：c

「try」以下のブロックで例外が発生すると、「except」までプログラムの制御が飛び、その後のブロックが実行されます。

Q 11-2 の解答

答え：c

Python では文法エラーを含むすべてのエラーが「例外」として扱われるので **a** は間違いです。例外が発生し、捕捉されないときはプログラムの実行は止まるので **c** は正解です。そのときに表示されるエラーメッセージなどを「トレースバック」と言います。また、複数の例外が重なって表示されることから「スタックトレース」などとも呼ばれますが、ストックとは呼ばれません。**d** は少し難しいかもしれませんが、例外はエラーの目的以外で、プログラムの流れを変える「フロー制御」に使われることがあります。

Q 11-3 の解答

答え：b

Python で例外を発生するには「raise 文」を使います

Q 11-4 の解答

答え：b、d

with 文は末尾にコロン (:) があり、直後にインデントブロックを伴うので **a** は正しいです。ファイル操作だけでなく、Chapter14 で見るように他のリソース確保に関わる処理にも使われるので **b** は間違いです。**c** は正しく、リソースの確保と解放を暗黙に実行してくれるため **d** は間違いになります。

Q 11-5 の解答

答え：1:c、2:d、3:a

本書の解説では、プログラムの流れを変えるフロー制御という文脈で、finally 句と with 文について同じパートで解説しました。その内容を確認する問題です。空欄 1 には選択肢の中で唯一例外にかかわる「finally」が入ります。空欄 2 は b と迷う人がいるかもしれませんが、本書の解説から「リソース」を選んで欲しいところです。空欄 3 は「リソースの操作を暗黙に実行できる」という内容から「with 文」となります。

Q 11-6 の解答

答え：1:b、2:c

try ~ except の書き方に関する問題です。空欄 1 には例外を捕捉したいブロックの前に置く「try」が入ります。except の後には「例外クラス」が置かれるため、この選択肢の中で唯一の例外クラス「ValueError」を選びます。

ChapterX (総合)

Q X-1 の解答

答え：2 行目の「6」を「len(steps)」にする

2 行目の「6」を「7」にしてもよいのですが、len() 関数でリストの要素数を得る方法の方が、より抽象的で好ましい修正です。

Q X-2 の解答

答え：5 行目を「return random.choice(items)」にする

omikuji() 関数からの戻り値を受け取るようになっているのに、return がないことが原因で期待通りの動きになっていません。関数内の print() の行 (5 行目) を「return random.choice(items)」のようにして、文字列を戻り値で返すように修正すれば、期待通りの動きになります。

Q X-3 の解答

答え：13 行目を「words = [word for word **not** in words if word in {"the", "and", "is"}]」にする

プログラムを実際に動かしてみると、取り除いたはずの「the」「and」「is」のみが数えられています。そのことから、13 行目のリスト内包表記が間違っていることが類推できるはずです。if の条件をよく見ると「in」となっており、この3つの単語のみが結果としてリストに入っていることがわかります。この「in」を「not in」にすると期待通りの結果になります。