アルゴリズム入門　共通問題（2024年度Aセメスター試験）

試験日時：2025年1月30日第4限（60分）　答案用紙：1部　持ち込み一切不可

裏面にも問題があるので確認すること

内容に関する質問は受け付けない．問題の記述が曖昧な場合は，適切な仮定をおいて回答し，どのような仮定をおいたのか明記せよ．

`make1d(n)` はすべての要素が0で長さが`n`の配列を作る関数であり，定義済みだとする．また，各基本的操作やライブラリ関数の呼出の計算量は $O(1)$ だとする．

問題1

賭けを繰り返すときの所持金額の推移を考える．最初`x_0`枚のコインを所持しており，賭けを1回行うごとに確率`theta`で1枚のコインを獲得し，確率$1-$`theta`で1枚のコインを失うとする．

(1) `random.random`関数は0以上1未満の小数値をランダムに生成する．確率`theta`で1，$1-$`theta`で-1を返す右記の関数`Bernoulli`を空欄[A]・[B]を埋めることで完成させよ．

(2) 所持するコインが0枚になっても賭けを継続し，負の枚数のコインを所持できるとする．このとき，T回の賭けを繰り返したときの所持コイン枚数の推移を長さ$T+1$の配列として返す右記の関数`simulate_without_ruin`を，関数`f`を5行以内のプログラムで定義することで完成させよ．

(3) 所持するコインが0枚になった時点で賭けは中断し，破産するとする．このとき，T回目の賭けの終了時点あるいはそれまでに破産する確率をモンテカルロ法を用いて近似的に求めることを考える．10000回のシミュレーションで破産した回数を求める右記の関数`count_ruin`を空欄[C]・[D]を埋めることで完成させよ．また，`count_ruin`をある入力で実行した結果が550であったなら，破産する確率はおよそいくつだと考えられるか．

(4) `random.random`関数が返す値は真にランダムなものとは区別して擬似乱数と呼ばれる．擬似乱数がもつと期待される性質を2つ挙げて説明せよ．

```
import random

def Bernoulli(theta):
  x = random.random()
  if x < theta:
    return [ A ]
  else:
    return [ B ]

def simulate_without_ruin(x_0, theta, T):
  x = make1d(T + 1)
  x[0] = x_0
  for t in range(0, T):
    f(x, t, Bernoulli(theta))
  return x

def count_ruin(x_0, theta, T):
  c = 0
  for i in range(0, 10000):
    t = 1
    x = make1d(T + 1)
    x[0] = x_0
    [ C ]:
      f(x, t - 1, Bernoulli(theta))
      t = t + 1
    if x[t - 1] > 0:
      c = c + 1
  return [ D ]
```

問題 2

数値が小さい順に並んだ配列と目標値が与えられたとき，目標値未満で最大の配列要素が配列の何番目にあるかを求めるプログラムを作成したい．例えば配列 `[1,2,5,6,8,11,13]` と目標値 10 が与えられた場合，目標値未満の最大要素は 4 番目の 8 なので，4 を返す（先頭要素を 0 番目とする）．なお，目標値は配列の最小要素より大きいものとする．

右に示す関数 `search` は，配列 `x` と目標値 `k` を受け取ると，二分探索を用いて，目標値にちょうど等しい配列要素がなければ，目標値未満の最大要素の位置を求める．

```
def search(x, k):
  st = 0
  ed = len(x)
  while ed - st > 1:
    c = (st + ed) // 2
    if k < x[c]:
      [ A ] = c
    else:
      [ B ] = c
  return [ C ]
```

(1) 空欄 A ～ C を埋めて関数 `search` を完成させよ．

(2) `search([4,5,5,7], 6)` の実行において，`while` ループの条件 `ed - st > 1` を実行する際の `st` と `ed` の値はどのように変化し，最終的に関数 `search` は何を返すか．

(3) 入力配列の長さ n に対する関数 `search` の漸近時間計算量として最も厳密な見積りを以下から選べ．さらに，その理由を簡単に説明せよ．

$\mathrm{O}(n)$　$\mathrm{O}(n^2)$　$\mathrm{O}(n^3)$　$\mathrm{O}(2^n)$　$\mathrm{O}(\log n)$　$\mathrm{O}(n \log n)$　$\mathrm{O}(\log^2 n)$　$\mathrm{O}((\log n)^2)$

(4) 関数 `search` に配列 `x` と目標値 `k` を与えて実行したとき，`x[st]` と `k` が，また `ed` $<$ `len(x)` のときに `x[ed]` と `k` が，それぞれ `while` ループでの繰り返し中を通して満たす関係を不等式で表せ．できる限り厳しい条件を表す不等式とすること（例えば $x \le y$ より $x + 1 \le y$ の方がより厳しい）．なお，`x` に `k` と等しい要素はないとしてよい．

(5) 関数 `search` に配列 `x` と目標値 `k` を与えて実行した結果が n だったとき，`x` の n 番目はどのような要素なのかを 1 行程度でできる限り詳しく説明せよ．`x` が目標値と等しい値を含む場合・同じ値を複数含む場合を考慮に入れること．

(6) 関数 `search` のある 1 行だけを修正し，目標値に等しい配列要素がある場合でも正しい結果を求めるようにしたい．どの行をどのように修正すれば良いか．

問題 3

関数 $f(x)$ の数値積分 $\int_a^b f(x)dx$ について考える．図 1 のように区間 $[a, b]$ を n 個の幅 $d = \frac{b-a}{n}$ の微小区間に分割して，各区間を台形領域（図の灰色領域）で近似する．積分の近似値は，台形領域の面積 $\frac{(f(x_i)+f(x_{i+1}))d}{2}$ の総和として，次式のように求まる．

$$
\begin{aligned}
\int_a^b f(x)dx &\simeq \sum_{i=0}^{n-1} \frac{(f(x_i) + f(x_{i+1}))d}{2} \\
&= \left(f(x_0) + \sum_{i=1}^{n-1} f(x_i) + \sum_{i=1}^{n-1} f(x_i) + f(x_n) \right) \frac{d}{2} \\
&= \left(\frac{f(x_0)}{2} + \sum_{i=1}^{n-1} f(x_i) + \frac{f(x_n)}{2} \right) d \\
&= \left(\frac{f(a)}{2} + \sum_{i=1}^{n-1} f(a + d \cdot i) + \frac{f(b)}{2} \right) d
\end{aligned}
$$

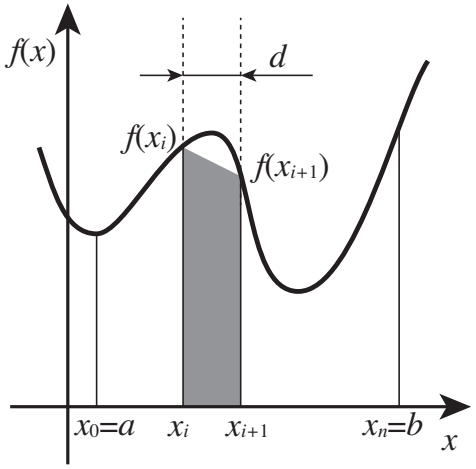

図 1: 台形近似による数値積分

(1) 図 2 に示す関数 trapezoid は関数 $f(x)$ を区間 $[a, b]$ で n 個の微小区間に分割して積分する．前記の説明を参考にして，空欄 A ～ D を埋めよ．

(2) 関数 $f(x) = 9x^8$ として，区間 $[-1, 0]$ と区間 $[0, 1]$ で積分する．なお，$f(x)$ は $x = 0$ に関して対称で下に凸な関数であり，$\int_{-1}^{0} f(x)dx = \int_{0}^{1} f(x)dx = 1$ である．

図 2 のプログラムに図 3 のプログラムを加えて実行してみたところ，実行結果は図 4 であった．これらの計算結果と理論値との違い（アルゴリズム自体に由来する誤差や，丸め誤差，桁落ち，情報落ちなど）について考える．

(a) 分割数が 10^1 から 10^6 の範囲において，分割数を多くする（微小区間を小さくする）ことによる誤差への影響とその理由について 2 行程度で述べよ．

(b) 分割数が 10^1 から 10^6 の範囲において，すべての誤差に共通する性質（台形近似による影響）とその理由について 2 行程度で述べよ．

(c) 分割数が $10^7, 10^8$ において，分割数がそれら未満の場合から予測される誤差の大きさとは異なっている点を指摘し，その理由として考えられることを 2 行程度で述べよ．

(d) 分割数が 10^1 から 10^6 では積分区間 $[-1, 0]$ と積分区間 $[0, 1]$ の誤差の大きさはおおむね等しい．しかし，分割数が $10^7, 10^8$ においては，積分区間 $[-1, 0]$ の誤差は $[0, 1]$ にくらべてかなり大きい．この違いをもたらした要因を挙げよ．また，その要因によってこの違いがもたらされたと推測できる根拠を 2 つ以上挙げよ．

```
def trapezoid(a, b, n):
  d = (  A  ) / n
  s = f(a) / 2
  for i in range(1,  B  ):
    s = s + f(  C  )
  return  D
```

図 2: 関数 trapezoid

```
def f(x):
  return 9 * (x ** 8)

for i in range(1, 9):
  n = 10 ** i
  print('10 **', i)
  print('  ', trapezoid(-1.0, 0.0, n))
  print('  ', trapezoid( 0.0, 1.0, n))
```

図 3: 分割数を変えながら区間 $[-1, 0]$ と区間 $[0, 1]$ について $f(x) = 9x^8$ を積分するプログラム

```
10 ** 1
  1.0595819970000004
  1.0595819970000002
10 ** 2
  1.0005999580020002
  1.0005999580020002
10 ** 3
  1.0000059999958
  1.0000059999958009
10 ** 4
  1.0000000599999967
  1.00000006
10 ** 5
  1.0000000005999732
  1.0000000006000056
10 ** 6
  1.000000000005818
  1.0000000000060096
10 ** 7
  0.999999999996213
  1.0000000000000764
10 ** 8
  0.9999999999604392
  1.0000000000000446
```

図 4: 図 3 のプログラムの実行結果