1 アルゴリズムとプログラミング

問70 Check □ □ □ 【オリジナル問題】

次のプログラム中の［ a ］と［ b ］に入れる正しい答えの組合せを，解答群の中から選べ。

関数simpsonMethodは，図１のように，曲線“y ＝ f(x)”とｘ軸の区間[a, b]に挟まれた領域の面積を，台形公式を使って求める。指定された区間[a, b]をｎ等分した各区間を台形と見なすと，各台形の面積（S_1，S_2，･･･，S_n）の和が，求めたい領域の面積Ｓにほぼ等しくなる。台形分割（４分割）の例を，図２に示す。

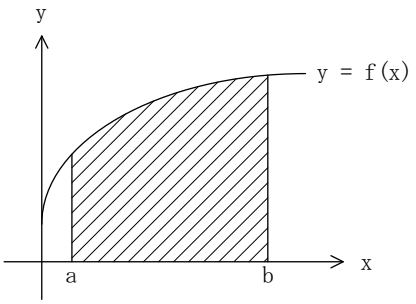

図１　求める領域の面積

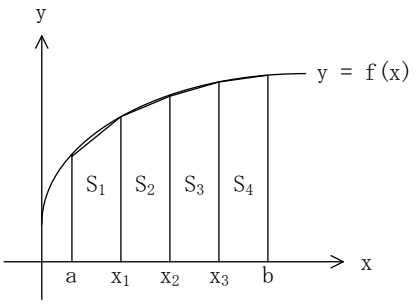

図２　台形分割の例（４分割）

台形分割の考え方から，次のように台形公式の一般形が求められる。

$$S = S_1 + S_2 + \cdots + S_n$$
$$= h \times \{f(a) \div 2 + f(x_1) + f(x_2) + \cdots + f(x_{n-1}) + f(b) \div 2\}$$
（ただし，$h = (b - a) \div n$，$x_i = a + i \times h$）

関数simpsonMethodでは，収束の判定をするために，次の手順で処理を行う。

① 区間をｎ（＝2^k）等分して求めた面積をS_kとする。

② S_0の面積{(f(a) ＋ f(b)) × (b － a) ÷ 2}を最初に求める。

③ S_k（k ＝ 1, 2, 3, ･･･）を順次求め，“$|S_k - S_{k-1}| <$ eps（収束値）”となった時点で計算を終了し，そのときのS_kを求める面積とする。

なお，関数simpsonMethodでは，引数で与えられたｘ座標に対する曲線上のｙ座標の値を返す関数fを使用する。

〔プログラム〕

```
大域: 実数型: eps          /* 収束値が格納されている */

/* 曲線“y ＝ f(x)”とx軸の区間[a, b]に挟まれた領域の面積を返す */
○実数型: simpsonMethod(実数型: a, 実数型: b)
  実数型: s1, s2, x
  整数型: i, n
  s1 ← 0
  s2 ← (f(a) ＋ f(b)) × (b － a) ÷ 2
  n ← 1
  while ((s2 － s1)の絶対値 が eps 以上)
    s1 ← s2
    n ← n × 2
    s2 ← (f(a) ＋ f(b)) ÷ 2
    for (i を 1 から (n － 1) まで 1 ずつ増やす)
      x ← [ a ]
      s2 ← s2 ＋ f(x)
    endfor
    s2 ← [ b ]
  endwhile
  return s2
```

解答群

	a	b
ア	((n － 1) × a ＋ i × b) ÷ n	s2 × ((b － a) ÷ n)
イ	((n － 1) × a ＋ i × b) ÷ n	s2 ÷ ((b － a) ÷ n)
ウ	((n － i) × a ＋ i × b) ÷ n	s2 × ((b － a) ÷ n)
エ	((n － i) × a ＋ i × b) ÷ n	s2 ÷ ((b － a) ÷ n)
オ	(a ＋ (i － 1) × b) ÷ n	s2 × ((b － a) ÷ n)
カ	(a ＋ (i － 1) × b) ÷ n	s2 ÷ ((b － a) ÷ n)
キ	(a ＋ i × b) ÷ n	s2 × ((b － a) ÷ n)
ク	(a ＋ i × b) ÷ n	s2 ÷ ((b － a) ÷ n)

1 アルゴリズムとプログラミング

問71 Check □ □ □ 【オリジナル問題】

次のプログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，配列の要素番号は0から始まる。

オートマトンは，入力に対する処理が内部の状態によって異なることをモデル化する表記法である。特に，有限個の状態と遷移で構成される有限オートマトンは，状態遷移図などを使用して表現される。

関数automatonは，任意の長さのビット列bitStrを引数として受け取り，図1の状態遷移図で表現される有限オートマトンに入力した結果を求める。結果は，ビット列が受理されれば1を，受理されなければ（－1）を返す。

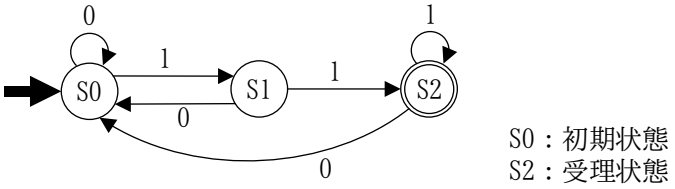

図1　状態遷移図

なお，ビット列bitStrは0又は1が格納された整数型の配列で表現し，先頭要素（bitStr[0]）から順に有限オートマトンに入力されるものとする。このとき，“配列の要素数＝ビット列のビット数”である。

また，図1の状態遷移図における内部の状態（S0～S2）と入力ビット（0，1）に対する遷移先（S0～S2）は，整数型の二次元配列transitionで表現する。

〔プログラム〕

```
○整数型: automaton(整数型の配列: bitStr)
  整数型の二次元配列: transition ← {{0, 1}, {0, 2}, {0, 2}}
  整数型: i, state
  state ← 0
  for (i を 0 から (bitStrの要素数 − 1) まで 1 ずつ増やす)
    state ← [　　]
  endfor
  if (state が 2 と等しい)
    return 1
  else
    return (−1)
  endif
```

解答群

ア　i ＋ transition[bitStr[i], state]

イ　i ＋ transition[state, bitStr[i]]

ウ　state ＋ transition[bitStr[i], state]

エ　state ＋ transition[state, bitStr[i]]

オ　transition[bitStr[i], state]

カ　transition[state, bitStr[i]]