問11　次のJavaプログラムの説明及びプログラムを読んで，設問に答えよ。

（Javaプログラムで使用するAPIの説明は，この冊子の末尾を参照してください。）

〔プログラムの説明〕

ギャップバッファを利用した簡易テキストエディタである。

ギャップバッファとは，編集対象となる文字列の編集箇所に空き領域を作り，そこに文字を挿入する機構をもつバッファのことである。一般に，テキストの編集時には，文字の挿入，削除などの変更は局所的に集中することが多く，編集箇所にあらかじめ空き領域を作っておくと効率よく変更を行うことができるという利点がある。ギャップバッファ内の空き領域をギャップといい，ギャップバッファ内の文字列をテキストという。ここで，ギャップは，ギャップバッファ内に一つしか存在しない。ただし，文字の挿入によって，ギャップがない状態になる場合がある。

テキストの表示イメージとギャップバッファを図1に示す（網掛けの部分がギャップである）。

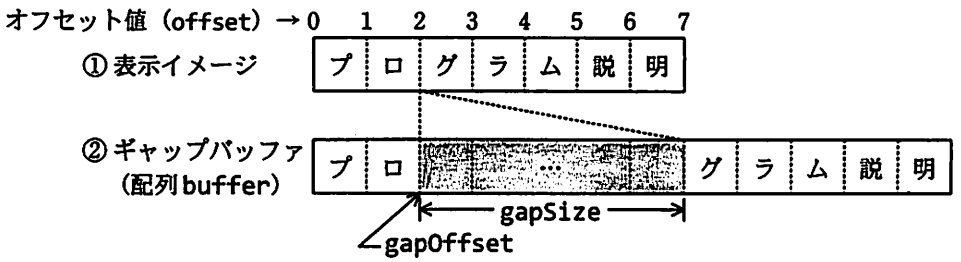

図1　テキストの表示イメージとギャップバッファ

クラス GapBuffer はギャップバッファの機能を実現する。このクラスの使用者は，テキストを仮想的に連続する文字列として扱うことができ，ギャップの位置やサイズを意識せずに文字の挿入や削除の処理を行う。このとき，図1①のように各文字はオフセット値（offset）で指定し，テキストの最初の文字はオフセット値0，テキストの最後の文字はオフセット値“テキストの文字数－1”で指定する。

図1②は，配列 buffer 内におけるテキストの物理的な表現の例である。フィールド gapOffset と gapSize は，それぞれギャップの先頭位置とギャップのサイズを表す。

図1②の状態から“プログラム”と“説明”の間に“の”を挿入する手順を図2に示す。

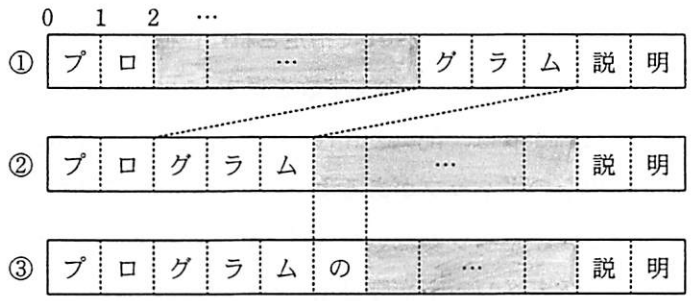

図2　挿入する手順

まず，“プロ”の直後に“グラム”を移動することで，ギャップを“プログラム”と“説明”の間に移動する（図2②）。ギャップの先頭に“の”を格納する（図2③）。この操作によって，ギャップのサイズは1文字分小さくなり，ギャップの先頭位置は1文字分後ろへずれる。

次に，図2③の状態から“プログラムの”の“グ”を削除する手順を図3に示す。

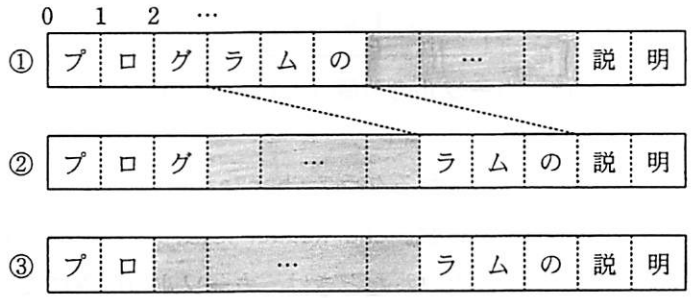

図3　削除する手順

まず，“ラムの”を“説明”の直前に移動することで，ギャップを“プログ”と“ラムの説明”の間に移動する（図3②）。ギャップの先頭位置を1文字分前にずらし，ギャップのサイズを1文字分増やす（図3③）。

クラス`GapBuffer`は，次のコンストラクタ及びメソッドをもつ。

(1) コンストラクタは引数`initialText`で指定された文字列をテキストの初期値としてギャップバッファを生成する。ギャップは，テキストの前に作られる。

(2) メソッド`insert`は，メソッド`confirmGap`を呼んで引数`offset`の位置にギャップを移動した上で，引数`offset`で指定された位置に，引数`ch`で指定された文字を挿入する。

(3) メソッド`delete`は，引数`offset`で指定された位置にある文字を削除する。テキストがない場合は，何もしない。

(4) メソッド charAt は，引数 offset で指定された位置の文字を返す。

(5) メソッド length は，テキストの文字数を返す。

(6) メソッド confirmGap は，配列 buffer において引数 newGapOffset で与えられた位置に 1 文字以上のギャップがあるようにする。必要であればギャップを指定された位置に移動し，ギャップがない場合は，新たにギャップを作る。

なお，このクラスの各メソッドが呼び出されるとき，引数は正しいものとする。

クラス Editor は，GapBuffer を利用して，簡易テキストエディタを実現する。メソッド main に与えられた最初の引数をテキストの初期値とする。カーソルは，文字の間にあり，カーソルの位置はオフセット値で表し，フィールド cursor に保持される。例えば，図 4 のようにカーソルが“プロ”と“グラム”の間にあるとき，cursor の値は 2 である。カーソルがテキストの末尾にあるとき，cursor の値はテキストの文字数と同じであり，テキストがないとき，cursor の値は 0 である。

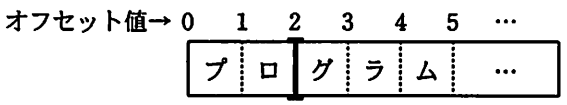

図4　カーソルの位置（cursor の値は 2）

文字の入力は，外部で与えられるクラス CharReader で行う。メソッド get は，ターミナルから 1 文字ずつ読み込む。CharReader には，次の制御文字が定義されている。制御文字以外の文字が入力されたときは，カーソルの位置に文字を挿入し，カーソルを 1 文字分進める。

① CharReader.MOVE_FORWARD

カーソルを 1 文字分進める。カーソルがテキストの末尾にあるときは，無視する。

② CharReader.MOVE_BACKWARD

カーソルを 1 文字分戻す。カーソルがテキストの先頭にあるときは，無視する。

③ CharReader.DELETE

カーソルの直後の 1 文字を削除する。カーソルがテキストの末尾にあるときは，無視する。

④ CharReader.EOF

エディタを終了する。

ギャップバッファの内容の表示は，外部で与えられるクラス Display で行う。メソッド output は，GapBuffer のメソッド charAt を呼び出してギャップバッファのテキストを取得する。

〔プログラム1〕

```
class GapBuffer {
   private static final int INITIAL_GAP_SIZE = 128;
   private char[] buffer;
   private int gapOffset = 0;
   private int gapSize = INITIAL_GAP_SIZE;

   GapBuffer(String initialText) {
      buffer = new char[initialText.length() + gapSize];
      System.arraycopy(initialText.toCharArray(), 0,
                       buffer, gapSize, initialText.length());
   }

   void insert(int offset, char ch) {
      confirmGap(offset);
      buffer[gapOffset++] = ch;
      [   a   ];
   }

   void delete(int offset) {
      if (length() == 0)
         return;
      confirmGap(offset + 1);
      gapOffset--;
      gapSize++;
   }

   char charAt(int offset) {
      if (offset >= gapOffset)
         offset += [   b   ];
      return buffer[offset];
   }

   int length() { return [   c   ]; }

   private void confirmGap(int newGapOffset) {
      if (gapSize == 0) {
         char[] temp = new char[buffer.length + INITIAL_GAP_SIZE];
         System.arraycopy(buffer, 0, temp, 0, buffer.length);
         gapOffset = buffer.length;
         gapSize = INITIAL_GAP_SIZE;
         buffer = temp;
      }
      if (newGapOffset < gapOffset) {
         System.arraycopy(buffer, newGapOffset, buffer,
                newGapOffset + gapSize, gapOffset - newGapOffset);
      } else {
         System.arraycopy(buffer, gapOffset + gapSize,
                buffer, gapOffset, newGapOffset - gapOffset);
      }
```

```
      gapOffset = newGapOffset;
    }
  }
```

〔プログラム2〕

```
class Editor {
    private [ d ] buf;
    private int cursor = 0;

    private Editor(String text) {
        buf = new GapBuffer(text);
    }

    private void run() {
        Display.output(buf, cursor);
        char ch;
        while ((ch = CharReader.get()) != CharReader.EOF) {
            switch (ch) {
            case CharReader.MOVE_FORWARD:
                moveCursor(1);
                break;
            case CharReader.MOVE_BACKWARD:
                moveCursor(-1);
                break;
            case CharReader.DELETE:
                if (cursor < buf.length()) {
                    buf.delete([ e ]);
                }
                break;
            default:
                buf.insert([ f ], ch);
                break;
            }
            Display.output(buf, cursor);
        }
    }

    private void moveCursor(int n) {
        int newCursor = cursor + n;
        if (newCursor >= 0 && newCursor <= buf.length()) {
            cursor = newCursor;
        }
    }

    public static void main(String[] args) {
        Editor editor = new Editor(args[0]);
        editor.run();
    }
}
```

設問　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。

aに関する解答群

ア　gapSize--　　イ　gapSize++

ウ　gapSize - 1　　エ　gapSize + 1

bに関する解答群

ア　buffer.length　　イ　gapOffset

ウ　gapSize　　エ　length()

cに関する解答群

ア　buffer.length

イ　buffer.length - gapOffset

ウ　buffer.length - gapSize

エ　buffer.length - INITIAL_GAP_SIZE

dに関する解答群

ア　CharReader　　イ　Display　　ウ　Editor

エ　GapBuffer　　オ　Object　　カ　StringBuffer

e，fに関する解答群

ア　--cursor　　イ　++cursor　　ウ　cursor

エ　cursor--　　オ　cursor++