영어로 자동변환을 위한 데이터 추출

입력모드가 한글인 상태로 타이핑을 하고, 이것이 영어단어일 경우에 영어로 변경하기 위한 데이터 추출 프로그램을 작성합니다.

```
ab
aba
abaca
abaci
aback
abacus
abaft
abalone
abandon
abandoned
abandoner
```

절대 한글로 칠 수 없는 단어도 있지만, 한글과 겹치는 단어들도 존재합니다. 이를 구별할 필요가 있습니다.

snippet.java

```
    HashMap<String, Integer> mWords = new HashMap<String, Integer>();
    List<WordInfo> mWordList = new ArrayList<WordInfo>();
    List<String> mMoumList = new ArrayList<String>();
    List<String> mDoubleJaum = new ArrayList<String>();
    List<String> mDoubleMoum = new ArrayList<String>();
    private boolean mBroken = false;
    private int mRearWordCount = 0;
    private boolean mBrokedHistory;;

      public static void main(String[] args) {
        MakeCode mc = new MakeCode();
        mc.start();
    }
```

필요 변수를 선언하고 객체를 생성합니다.

snippet.java

```
    public MakeCode() {
        String moums = "yYuUiIoOpPhHjJkKlLbBnNmM";
        String moum = "";

        for (int i = 0; i < moums.length(); i++) {
            moum = moums.substring(i, i + 1);
            mMoumList.add(moum);
```

```
        }

        mDoubleJaum.add("rt"); // ㄲ
        mDoubleJaum.add("sw"); // ㄵ
        mDoubleJaum.add("sg"); // ㄶ
        mDoubleJaum.add("fr"); // ㄺ
        mDoubleJaum.add("fa"); // ㄻ
        mDoubleJaum.add("fq"); // ㄼ
        mDoubleJaum.add("ft"); // ㄽ
        mDoubleJaum.add("fx"); // ㄾ
        mDoubleJaum.add("fv"); // ㄿ
        mDoubleJaum.add("fg"); // ㅀ
        mDoubleJaum.add("qt"); // ㅃ

        mDoubleMoum.add("hk"); // ㅘ
        mDoubleMoum.add("ho"); // ㅙ
        mDoubleMoum.add("hl"); // ㅚ
        mDoubleMoum.add("nj"); // ㅝ
        mDoubleMoum.add("np"); // ㅞ
        mDoubleMoum.add("nl"); // ㅟ
        mDoubleMoum.add("ml"); // ㅢ

    }
```

이중모음, 이중자음에 대한 경우를 입력합니다. 대문자 입력에 대한 경우가 포함되지 않았습니다. (≪ 수정해야 할 사항)

snippet.java

```
    private void start() {
        // TODO Auto-generated method stub

        // read
        BufferedReader br = null;
        try {
            br = new BufferedReader(new FileReader("words.txt"));

            StringBuilder sb = new StringBuilder();
            String line = br.readLine();

            while (line != null) {

                boolean bAdd = false;

                int leng = getKoreanLength(line);

                if (hasBroken() || leng >= 3){
                    bAdd = true;
                }
                else {
```

```
                    line = br.readLine();
                    continue;
                }

                mWordList.add(new WordInfo(line, leng));
                mWords.put(line,leng);

                // 앞글자 대문자 변환
                line = Character.toUpperCase(line.charAt(0))
                        + line.substring(1);
                leng = getKoreanLength(line);
                mWordList.add(new WordInfo(line, leng));
                mWords.put(line,leng);

                line = br.readLine();
            }
        } catch (Exception e) {

        } finally {
            try {
                br.close();
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }

        // write
        try {
            //
////////////////////////////////////////////////////////////
            BufferedWriter out = new BufferedWriter(new
FileWriter("out.txt"));

            for (WordInfo wi : mWordList) {
                String key = wi.mWord;
                int cnt = wi.mCnt;

                out.write(":*:"+key+"::\n");
                out.write("        E("+cnt+")\n");
                out.write("        return\n");

            }

//            for (Entry<String, Integer> entry : sortedMap.entrySet())
{
//                String key = entry.getKey();
//                Integer cnt = entry.getValue();
//
//                out.write(":*:"+key+"::\n");
//                out.write("        WRITE_IN_ENGLISH(\""+key+"\",
```

```
"+cnt+")\n");
//                out.write("        return\n");
//            }

//            String s = "출력 파일에 저장될 이런 저런 문자열입니다.";
//
//            out.write(s);
//            out.newLine();
//            out.write(s);
//            out.newLine();

            out.close();
            //
////////////////////////////////////////////////////////////
        } catch (IOException e) {
            System.err.println(e); // 에러가 있다면 메시지 출력
            System.exit(1);
        }

    }
```

단어 파일을 읽어서, 하나씩 분석하고, 파일에 저장합니다.

snippet.java

```
    public int getKoreanLength(String line) {
        mBrokedHistory = false;

        // 기존에 변경된 문자가 있는가?
        int longest = getLongestSameWord(line);

        // 기중에서 가장 긴 문자, 이후의 한글 수를 찾아야 함.
        String korWord = line.substring(longest);

        mRearWordCount = 0;

        String lastChar = "";

        STATE next = STATE.CHOSUNG;
        for (String stroke : korWord.split("")) {
            if (next == STATE.CHOSUNG) {
                if (isMoum(stroke)) {
                    next = STATE.JUNGSUNG_DOUBLE;
                    setBroken();
                } else {
                    next = STATE.CHOSUNG_DOUBLE;
                }
            } else if (next == STATE.CHOSUNG_DOUBLE) {
                if (isMoum(stroke)) {
                    next = STATE.JUNGSUNG_DOUBLE;
```

```
                } else if (isDoubleJaum(lastChar, stroke)) {
                    next = STATE.JUNGSUNG;
                } else {
                    setBroken();
                    addLength();
                    next = STATE.CHOSUNG_DOUBLE;
                }
            } else if (next == STATE.JUNGSUNG_DOUBLE) {
                if (isDoubleMoum(lastChar, stroke)) {
                    if (isBroken()) {
                        next = STATE.CHOSUNG;
                        addLength();
                    } else {
                        next = STATE.JONGSUNG;
                    }
                } else if (isMoum(stroke)) {
                    addLength();
                    setBroken();
                    next = STATE.JUNGSUNG_DOUBLE;
                } else {
                    if (isBroken()) {
                        addLength();
                        next = STATE.CHOSUNG_DOUBLE;
                    } else {
                        next = STATE.JONGSUNG_DOUBLE;
                    }
                }
            } else if (next == STATE.JUNGSUNG) {
                if (isMoum(stroke)) {
                    next = STATE.JUNGSUNG_DOUBLE;
                } else {
                    addLength();
                    next = STATE.JUNGSUNG;
                }
            } else if (next == STATE.JONGSUNG) {
                if (isMoum(stroke)) {
                    addLength();
                    setBroken();
                    next = STATE.JUNGSUNG_DOUBLE;
                } else {
                    next = STATE.JONGSUNG_DOUBLE;
                }
            } else if (next == STATE.JONGSUNG_DOUBLE) {
                if (isMoum(stroke)) {
                    addLength();
                    next = STATE.JUNGSUNG_DOUBLE;
                } else {
                    // 이전 값이 같이 쓸 수 있는 자음인가?
                    if (isDoubleJaum(lastChar, stroke)) {
                        next = STATE.CHOSUNG;
                        addLength();
```

```
                    } else {
                        next = STATE.JUNGSUNG;
                        addLength();
                    }
                }
            }

            lastChar = stroke;
        }

        if (next != STATE.CHOSUNG) {
            addLength();
        }

        System.out.println(line + "\tSplit:" + longest + ":" + korWord
                + "\tCNT:" + mRearWordCount);

        return longest + mRearWordCount;
    }

    private boolean isBroken() {
        return mBroken;
    }
```

각 글자 하나씩 읽어서, 한글일 경우의 상태값을 기록하면서 몇 글자로 입력이 되었는지 판단합니다.

관련 문서

- Java