Section: CV CPP
Student: Alex Baraian Project
Due date: 5/11/2023
Project Number: 8

**Algorithm Steps**

```
S1: open all files from argv[]
S2: thrVal=argv[2]
S3: read in numRows,numCols,minVal,maxVal from inFile
S4: read int numStructRows,numStructCols,StructMin,StructMax,rowOrigin,colOrigin from
structElemFIle
S5: loadImage (inFile, imgAry)
S6:output to outFile1 ," Below is the input image
S7: imgReformat (imgAry, outFile1)
S8:computePP (imgAry)
S9:output to outFile2  "Below is HPP"
S10:printPP (HPP, outFile2)
S11:output to outFile2  "Below is VPP"
S12:printPP (VPP, outFile2)
S13:: binaryThreshold (HPP, thrVal, binHPP)
S14:binaryThreshold (VPP, thrVal, binVPP)
S15:output to outFile2  "Below is binHPP"
S16: printPP (binHPP, outFile2)
S17:output to outFile2  "Below is binVPP"
S18:printPP (binVPP, outFile2)
S19:(boxNode*) zBox  computeZoneBox (binHPP, binVPP)
S20:istInsert (listHead, zBox)
S21:output to outFile2  "Below is the linked list after insert input zone box"
S22: printBox (listHead, outFile2)
S23:morphClosing (binHPP, structElem, morphHPP)
S24:morphClosing (binVPP, structElem, morphVPP)
S25:output to  outFile2  "Below is morphHPP after performing morphClosing on HPP"
S26:output to moutFile2  printPP (morphHPP)
S27:output to  outFile2  "Below is morphVPP after performing morphClosing on VPP"
S28:printPP (morphVPP)
S29:runsHPP  computePPruns (morphHPP, numRows)
S30:runsVPP  computePPruns (morphVPP, numCols)
S31: output to  outFile2  The number of runs in morphHPP-runsHPP is "
S32: output to outFile2  The number of runs in morphVPP – runsVPP is "
S33:readingDirection  computeDirection (runsHPP, runsVPP)
S34:outFile2  "readingDirection is" /
S35:: if readingDirection == 1
          computeHorizontalTextBox (zoneBox, morphHPP, numRows)
       else if readingDirection == 2
```

```
                    computeVerticalTextBox (zoneBox, morphVPP, numCols)
S36:overlayBox (listHead, imgAry)
S37:output to outFile1  “Below is the input image overlay with bounding boxes”
S38:imgReformat (imgAry)
S39:output to outFile1  “Output the boxNode in the list”
S40:printBox (listHead, outFile1)
S41: close files
```

**Source Files**

```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <algorithm>
#include <sstream>
#include <math.h>
using namespace std;
class boxNode{
    public:
        int boxType;
        int minR;
        int minC;
        int maxR;
        int maxC;
        boxNode* next;

        boxNode(int a,int b, int c, int d,int e, boxNode* n){
            boxType=a;
            minR=b;
            minC=c;
            maxR=d;
            maxC=e;
            next=n;
        }
};
class docImage{
    public:
        int thrVal;
        int numRows;
        int numCols;
        int minVal;
        int maxVal;
        int numStructRows;
        int numStructCols;
        int structMin;
        int structMax;
        int rowOrigin;
        int colOrigin;
        int** imgAry;
        int* structElem;
        int* HPP;
        int* VPP;
        int* binHPP;
        int* binVPP;
        int* morphHPP;
        int* morphVPP;
        boxNode* listHead;
        int runsHPP;
        int runsVPP;
        int readingDirection;

    //constructor
    void loadImage(ifstream& inFile,ifstream& structElemFile){
        structElem = new int[numStructRows];

        for(int row=0;row<numStructRows;row++){ //setting up structAry

                structElemFile>>structElem[row];

        }
        imgAry = new int*[numRows+2];
        for(int i=0;i<=numRows+1;i++){
            imgAry[i] = new int[numCols+2];
        }

        for(int row=0;row<=numRows+1;row++){
            for(int col=0;col<=numCols+1;col++){
                imgAry[row][col]=0;
            }
        }
```

```
        for(int row=1;row<=numRows;row++){  //setting up img ary
            for(int col=1;col<=numCols;col++){
                inFile>>imgAry[row][col];
            }
        }
        HPP= new int[numRows+2];
        VPP = new int[numCols+2];
        binHPP = new int[numRows+2];
        binVPP = new int[numCols+2];
        morphHPP = new int[numRows+2];
        morphVPP = new int[numCols+2];
        for(int i=0;i<=numRows+1;i++){
            HPP[i]=0;
            binHPP[i]=0;
            morphHPP[i]=0;
        }
         for(int i=0;i<=numCols+1;i++){
            VPP[i]=0;
            binVPP[i]=0;
            morphVPP[i]=0;
        }
    }
    void computePP(){
        for(int row=1;row<=numRows;row++){
            for(int col=1;col<=numCols;col++){
                if(imgAry[row][col]>0){
                    HPP[row-1]++;
                    VPP[col-1]++;
                }
            }
        }
    }
    void binaryThreshold(int reg[],int bin[],int size){
        for(int i=0;i<size;i++){
            if(reg[i]>=thrVal){
                bin[i]=1;
            }
        }
    }
    void printPP(int* a,ofstream& file,int size){
```

```
    for(int i=0;i<size;i++){
        file<<a[i]<<" ";
    }
    file<<endl;
}
boxNode* computeZoneBox(){
    int minR=1;
    int minC=1;
    int maxR=numRows;
    int maxC=numCols;
    while(binHPP[minR]==0 && minR<=numRows){
        if(binHPP[minR]==0){
            minR++;
        }
    }
    while(binHPP[maxR]==0 && maxR>=1){
        if(binHPP[maxR]==0){
            maxR--;
        }
    }
    while(binVPP[minC]==0 && minC<=numCols){
        if(binVPP[minC]==0){
            minC++;
        }
    }
     while(binVPP[minC]==0 && maxC>=1){
        if(binVPP[minC]==0){
            maxC--;
        }
    }

    boxNode* B = new boxNode(1,minR,minC,maxR,maxC,nullptr);
    return B;

}
void morphClosing(int* arr1,int* arr2,int size){
    int jOffset,cindex;
    bool match;
    int* temp = new int[size];
    for(int i=0;i<size;i++){
```

```
                        temp[i]=0;

        }
        for(int i=1;i<size;i++){
            if(arr1[i]==1){
                jOffset = i-colOrigin;
                while(jOffset<= i+colOrigin){
                    temp[jOffset]=1;
                    jOffset++;
                }
            }
        }

          for(int i=1;i<size;i++){


            if(temp[i]==1){
                match = true;
                            jOffset=i-colOrigin;
                while(match==true &&jOffset<=i+colOrigin){
                    if(temp[jOffset]==0){
                        match=false;
                    }
                    jOffset++;
            }
            if(match){
                arr2[i]=1;
            }
            else{
                arr2[i]=0;
            }
            }

        }

    }
    void lsitInsert(boxNode* a){
        a->next=listHead->next;
        listHead->next=a;
    }
```

```
int computePPruns(int* PP,int lastIndex ){
    int numRuns=0;
    int index=1;
    while(index<=lastIndex){
        while(PP[index]==0 && index<=lastIndex){
            if(PP[index]==0){
                index++;
            }
        }
        while(PP[index]>0 && index<=lastIndex){
            if(PP[index]>0){
                index++;
            }
        }
        numRuns++;
    }
    return numRuns;

}
void computeVerticalTextBox(boxNode* zBox){
    int minR = zBox->minR;
    int minC = zBox->minC;
    int maxR=zBox->maxR;
    int maxC=minC;
    int index=1;

    while(morphVPP[maxC]==0 && maxC<=numCols){
        if(morphVPP[maxC]==0){
            maxC++;
        }
    }

    minC=maxC;
    while(maxC<=numCols){
    while(morphVPP[maxC]>0 && maxC<=numCols){
        if(morphVPP[maxC]>0){
            maxC++;
        }
    }
    boxNode* B = new boxNode(2,minR,minC,maxR,maxC,nullptr);
    lsitInsert(B);
    minC=maxC;
    while(morphVPP[minC]==0 && minC<=numCols){
        if(morphVPP[minC]==0){
            minC++;
        }
    }
    maxC=minC;
    }
}
void computeHorizontalTextBox(boxNode* zBox){
    int minR = zBox->minR;
    int minC = zBox->minC;
    int maxR=minR;
    int maxC=zBox->maxC;
    int index=1;
    while(morphHPP[maxR]==0 && maxR<=numRows){
        if(morphHPP[maxR]==0){
            maxR++;
        }
    }
    minR=maxR;
    while(maxR<=numRows){
    while(morphHPP[maxR]>0 && maxR<=numRows){
        if(morphHPP[maxR]>0){
            maxR++;
        }
    }
    boxNode* B = new boxNode(2,minR,minC,maxR,maxC,nullptr);
    lsitInsert(B);
    minR=maxR;
    while(morphHPP[minR]==0 && minR<=numRows){
        if(morphHPP[minR]==0){
            minR++;
        }
    }
    maxR=minR;
    }
}
void computeDirection(ofstream& file){
    int factor =2;
    int direction=0;
    if(runsHPP<=2&&runsVPP<=2){
        file<<"The zone may be a non-text zone"<<endl;
    }
    else if(runsHPP>=factor*runsVPP){
        file<<"The document reading direction is horizontal!"<<endl;
        readingDirection=1;
    }
    else if(runsVPP>=factor*runsHPP){
        file<<"The document reading direction is vertical"<<endl;
        readingDirection=2;
    }
    else{
        file<<"The zone may be a non-text zone"<<endl;
    }
    readingDirection;
}
void imgReformat(int** a,ofstream &file){
        int max=0;
       for(int row=1;row<numRows+2;row++){
            for(int col=1;col<numCols+2;col++){
               if(max<a[row][col]){
                max=a[row][col];
               }

            }

        }


       // file<<numRows<<" "<<numCols<<" "<<min<<" "<<max<<endl;
        string str = to_string(max);
        int width = str.length(), r=1,c=1,ww;


        while(r<=numRows){
            c=1;
            while(c<=numCols){
                if(a[r][c]>0){
                file<<a[r][c];
                }
                else{
                    file<<".";
                }
                str= to_string(a[r][c]);
                ww=str.length();

                while(ww<=width){
                    file<<" ";
                    ww++;
                }

                c++;
            }
            file<<endl;
            r++;
        }
        file<<endl;


    }
void overlayBox(){
    boxNode* curr= listHead->next;
    int startR,endR,startC,endC,type;
    while(curr->boxType!=1){
        startC=curr->minC;
        endC=curr->maxC;
        startR=curr->minR;
        endR=curr->maxR;
        type=curr->boxType;
        for(int col=startC;col<=endC;col++){
            imgAry[startR][col]=type;
            imgAry[endR][col]=type;
        }
        for(int row=startR;row<=endR;row++){
            imgAry[row][endC]=type;
            imgAry[row][startC]=type;
        }
        curr=curr->next;
    }     }
void printBox(ofstream& file){
    boxNode* curr= listHead->next;
    while(curr->next!=nullptr){
        file<<curr->boxType<<endl;
        file<<curr->minR<<" "<<curr->minC<<" "<<curr->maxR<<"
"<<curr->maxC<<endl;
        curr=curr->next;
    }
}
};
int main(int argc,const char* argv[]){
ifstream inFile,structElemFile;
ofstream outFile1,outFile2;
int thrVal;

inFile.open(argv[1]);
thrVal=atoi(argv[2]);
structElemFile.open(argv[3]);
outFile1.open(argv[4]);
outFile2.open(argv[5]);

docImage documentImage;
documentImage.thrVal=thrVal;

inFile>>documentImage.numRows>>documentImage.numCols>>documentImage.minVal
>>documentImage.maxVal;

structElemFile>>documentImage.numStructRows>>documentImage.numStructCols>>
documentImage.structMin>>documentImage.structMax;
structElemFile>>documentImage.rowOrigin>>documentImage.colOrigin;

documentImage.loadImage(inFile,structElemFile); //STEP 1
outFile1<<"Below is the input image"<<endl;
documentImage.imgReformat(documentImage.imgAry,outFile1);

documentImage.computePP();  //STEP 2
outFile2<<"Below is HPP"<<endl;

documentImage.printPP(documentImage.HPP,outFile2,documentImage.numRows+2);
```

```
    outFile2<<"Below is VPP"<<endl;

documentImage.printPP(documentImage.VPP,outFile2,documentImage.numCols+2);


documentImage.binaryThreshold(documentImage.HPP,documentImage.binHPP,docum
entImage.numRows+2);  //STEP 3

documentImage.binaryThreshold(documentImage.VPP,documentImage.binVPP,docum
entImage.numCols+2);
    outFile2<<"Below is binHPP"<<endl;

documentImage.printPP(documentImage.binHPP,outFile2,documentImage.numRows+
2);
    outFile2<<"Below is binVPP"<<endl;

documentImage.printPP(documentImage.binVPP,outFile2,documentImage.numCols+
2);

    documentImage.listHead = new boxNode(0,0,0,0,0,nullptr);
    boxNode* zBox = documentImage.computeZoneBox();
    documentImage.lsitInsert(zBox);
    outFile2<<"Below is the linked list after insert input zone
box"<<endl;
    documentImage.printBox(outFile2);


documentImage.morphClosing(documentImage.binHPP,documentImage.morphHPP,doc
umentImage.numRows+2);

documentImage.morphClosing(documentImage.binVPP,documentImage.morphVPP,doc
umentImage.numCols+2);
    outFile2<<"Below is morphHPP, after performing morphCLosing on
HPP"<<endl;

documentImage.printPP(documentImage.morphHPP,outFile2,documentImage.numRow
s+2);
    outFile2<<"Below is morphVPP after performing morphClosing on
VPP"<<endl;
```

```
documentImage.printPP(documentImage.morphVPP,outFile2,documentImage.numCol
s+2);


documentImage.runsHPP=documentImage.computePPruns(documentImage.morphHPP,d
ocumentImage.numRows);

documentImage.runsVPP=documentImage.computePPruns(documentImage.morphVPP,d
ocumentImage.numCols);
    outFile2<<"The number of runs in morphHPP-runsHPP is
"<<documentImage.runsHPP<<endl;
    outFile2<<"The number of runs in morphVPP-runsVPP is
"<<documentImage.runsVPP<<endl;

    documentImage.computeDirection(outFile1);
    outFile2<<"readingDirection is
"<<documentImage.readingDirection<<endl;
    if(documentImage.readingDirection==1){

documentImage.computeHorizontalTextBox(documentImage.listHead->next);
    }
    else if(documentImage.readingDirection==2){

documentImage.computeVerticalTextBox(documentImage.listHead->next);

    }
    documentImage.overlayBox();
        documentImage.imgReformat(documentImage.imgAry,outFile1);

    documentImage.printBox(outFile2);
}
```

## outFile1 Zone1

Below is the input image

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . 1 1 1 . . . . . 1 . 1 . . . 1 . 1 1 1 . . . . 1 1 1 1 1 1 . . . . 1 1 1 1 . . . 1 1 1 . . .
. 1 1 . 1 1 1 . . . 1 1 . 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 1 1 . . . . 1 1 1 1 . . . 1 1 1 1 1 .
. . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . 1 1 1 1 . . . . 1 1 1 1 1 1 . . . 1 . 1 1 1 . . . . 1 1 1 1 .
. 1 1 1 . 1 1 . . . . . 1 1 1 . . . . 1 1 1 . . . . . . . 1 1 1 . . . . . . . 1 1 . . . . 1 1 1 1 .
. . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. 1 1 1 . . . . 1 1 . 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . 1 . . 1 1 1 . . .
. . 1 . 1 . . . 1 1 1 . 1 . . . 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 1 . .
. 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . . . . 1 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 1 1 . .
. . . . . . . . . 1 1 1 . . . . 1 1 1 1 1 . . . . . 1 1 1 . . . . . . 1 1 1 . . . . . . 1 1 1 1 . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . 1 1 1 . . . . 1 1 . 1 . . . . . 1 1 1 . . . . . . 1 1 1 . . . . 1 1 1 1 . . . . 1 1 1 . . .
. . 1 1 1 1 1 1 . . . . 1 1 1 . . . 1 . . 1 1 1 . . . 1 1 1 1 1 1 . . . . 1 1 1 1 1 1 . 1 1 1 . . .
. 1 1 . 1 1 1 1 . . 1 1 . 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . 1 . . . . 1 1 1 1 1 1 1 1 1 1 1 1 .
. . . 1 1 1 . 1 . . . 1 1 1 1 1 . . . 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 . 1 1 1 . . . . 1 1 1 . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . 1 . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. 1 1 1 . . . 1 1 1 . 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . 1 . . 1 1 1 . . .
. . 1 . 1 . . 1 1 1 1 . 1 . . . 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 . . . .
. 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . 1 1 1 1 1 . . . .
. . 1 1 1 . . 1 1 1 . . . . . . . . 1 1 1 . . . . . 1 1 1 . . . . 1 1 1 1 1 . . . . . 1 1 1 . . . .
. . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . 1 . . . . .
. . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . 1 . . . . . . . 1 1 1 . . . . . 1 1 1 . . . . . 1 1 1 . . . . . . . . . . . . . . . . . . .
. . . . 1 1 1 . . . . . 1 . 1 . . . 1 1 1 1 . . . . . 1 1 1 1 1 1 . . . . 1 1 1 1 . . . 1 1 1 . . .
. . 1 . 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . 1 . . . . 1 1 1 1 . . . 1 1 1 . . .
. . . 1 1 1 . 1 . . . 1 1 1 1 1 . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 . 1 1 1 . . . 1 1 1 1 . .
. . . . . . . . . . 1 1 1 . . . . . . . . . . . . . . . 1 1 1 . . . . . . 1 1 1 . . . 1 1 1 . . . .
. . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. 1 1 1 1 . . . 1 1 . 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . 1 . . 1 1 1 . . .
. 1 1 1 1 . . . 1 1 . 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . 1 . . 1 1 1 . . .
. . 1 . 1 . . . 1 1 1 . 1 . . . 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 . . . .
. 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . . . .
. 1 1 1 . . . . . 1 1 1 . . . . . . 1 1 1 . . . . . 1 1 1 . . . . . 1 1 1 . . . . . 1 1 1 . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
```

The document reading direction is horizontal!

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 . . . 1 1 1 . . . . . 1 . 1 . . . 1 . 1 1 1 . . . . 1 1 1 1 1 1 . . . . 1 1 1 1 . . . 1 1 1 . . 2
2 1 1 . 1 1 1 . . . 1 1 . 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 1 1 . . . . 1 1 1 1 . . . 1 1 1 1 1 2
2 . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . 1 1 1 1 . . . . 1 1 1 1 1 1 . . . 1 . 1 1 1 . . . . 1 1 1 1 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
. . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 1 1 1 . . . . 1 1 . 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . 1 . . 1 1 1 . . 2
2 . 1 . 1 . . . 1 1 1 . 1 . . . 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 1 . 2
2 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . . . . 1 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 1 1 . 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . . .
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 . . . 1 1 1 . . . . 1 1 . 1 . . . . . 1 1 1 . . . . . . 1 1 1 . . . . 1 1 1 1 . . . . 1 1 1 . . 2
2 . 1 1 1 1 1 1 . . . . 1 1 1 . . . 1 . . 1 1 1 . . . 1 1 1 1 1 1 . . . . 1 1 1 1 1 1 . 1 1 1 . . 2
2 1 1 . 1 1 1 1 . . 1 1 . 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . 1 . . . . 1 1 1 1 1 1 1 1 1 1 1 1 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . 1 . . . . . .
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 1 1 1 . . . 1 1 1 . 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . 1 . . 1 1 1 . . 2
2 . 1 . 1 . . 1 1 1 1 . 1 . . . 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 . . . 2
2 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . 1 1 1 1 1 . . . 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
. . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . 1 . . . . .
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 . . . 1 . . . . . . . 1 1 1 . . . . . 1 1 1 . . . . . 1 1 1 . . . . . . . . . . . . . . . . . 2
2 . . . 1 1 1 . . . . . 1 . 1 . . . 1 1 1 1 . . . . . 1 1 1 1 1 1 . . . . 1 1 1 1 . . . 1 1 1 . . 2
2 . 1 . 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . 1 . . . . 1 1 1 1 . . . 1 1 1 . . 2
2 . . 1 1 1 . 1 . . . 1 1 1 1 1 . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 . 1 1 1 . . . 1 1 1 1 . 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
. . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 1 1 1 1 . . . 1 1 . 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . 1 . . 1 1 1 . . 2
2 1 1 1 1 . . . 1 1 . 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . 1 . . 1 1 1 . . 2
2 . 1 . 1 . . . 1 1 1 . 1 . . . 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . 1 1 1 1 . . . 2
2 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . . . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 . . . . 1 1 1 . . . 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . . . . . . . . .
```

## outFile2 Zone1

```
Below is HPP
0 0 1 1 22 29 29 20 1 1 0 1 0 24 29 29 18 1 0 1 0 1 1 19 28 32 27 0 1 0 1 2 0 25 28 28 20 1 1 1 0 2 1 10 22 26 26 12 1 1 0 1 0 0 25 25 27 26 18 1 0 0
Below is VPP
0 11 18 14 25 11 9 8 11 13 12 17 17 9 12 10 8 11 16 17 17 12 9 5 8 10 13 19 24 23 16 10 18 9 13 13 17 20 17 12 10 7 7 11 23 24 18 9 4 0 0 0
Below is binHPP
0 0 0 0 1 1 1 1 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 0 0 0
Below is binVPP
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
Below is the linked list after insert input zone box
Below is morphHPP, after performing morphCLosing on HPP
0 0 0 0 1 1 1 1 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 0 0 0
Below is morphVPP after performing morphClosing on VPP
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
The number of runs in morphHPP-runsHPP is 7
The number of runs in morphVPP-runsVPP is 2
readingDirection is 1
2
54 1 59 50
2
43 1 48 50
2
33 1 37 50
2
23 1 27 50
```

```
2
13 1 17 50
2
4 1 8 50
```

## outFile1 Zone2

Below is the input image

```
..................................................
..1111..............111...........................
.11111...........1.11111.............1111.........
.11111.....1111....11111....11111...11111.........
..1111....11111....11111.....11.1...11111.........
.11111....11111....111......11111...1.111...1111..
............111.............1111.....1111...11111.
...........111..............1111............1111..
............................................11111.
..1111.............111.......................1111.
.11111.............11111............1111..........
.11111.1..11111....1111....1111.1...11111.........
.11111....11111....1111.....11111...11111.........
..1111....1111.....11111....11111....111....1111..
...........111..............111.....1111....11111.
............111..............1111............1111.
............................................11111.
.11111.............11111....................11111.
.11111.............11111.............1111.........
..1111....11111....111......11111...1.111.........
..1111....1.111....1111.....11111...11111.........
.1111.....11111.....1111.....1111....111....111.1.
..........1111..............1111....1.111...11111.
...........1111.............11111...........1111..
.............................................1111.
.11111............11111.....................1111..
.11111.............11111............11.11.........
..1111....1111......1111....11111...1111..........
.1111......1111....1111.....11111...11.111........
..1111....11111.....1111....11111...11.11...111.1.
..........11111..............111.....111.....1111.
..........11.11.............1111............111...
............................................11111.
.11111.............11111.....................1111.
.11111.........1...111..........1...1.111...11111.
..1111......111.....1111....1111....11111.........
.1111.....11111....1111......1111....1111.........
..1111....111......1111..1..1111....1111..........
.1111...1.1111......1111.....1111...11111....1111.
...........1111.............11111...1111.....111..
..........1111...............1111...........11111.
............................1111............11111.
............................................11111.
..................................................
..................................................
```

The document reading direction is vertical

```
222222...222222...222222...222222..222222..222222.
2.1112...2....2...2.1112...2....2..2....2..2....2.
211112...2....2..1211112...2....2..2.1112..2....2.
211112...2.1112...211112...211112..211112..2....2.
2.1112...211112...211112...2.11.2..211112..2....2.
211112...211112...2111.2...211112..21.112..211112.
2....2...2..112...2....2...211112..2.1112..211112.
2....2...2.1112...2....2...211112..2....2..211112.
2....2...2....2...2....2...2....2..2....2..211112.
2.1112...2....2...2111.2...2....2..2....2..2.1112.
211112...2....2...211112...2....2..211112..2....2.
211112.1.211112...211112...2111.2..211112..2....2.
211112...211112...211112...211112..211112..2....2.
2.1112...211112...211112...211112..2.1112..211112.
2....2...2.1112...2....2...2111.2..211112..211112.
2....2...2..112...2....2...2.1112..2....2..2.1112.
2....2...2....2...2....2...2....2..2....2..211112.
211112...2....2...211112...2....2..2....2..211112.
211112...2....2...211112...2....2..2.1112..2....2.
2.1112...211112...2111.2...211112..21.112..2....2.
2.1112...21.112...211112...211112..211112..2....2.
211112...211112...2.1112...2.1112..2.1112..2111.2.
2....2...211112...2....2...211112..21.112..211112.
2....2...2.1112...2....2...211112..2....2..211112.
2....2...2....2...2....2...2....2..2....2..2.1112.
211112...2....2...211112...2....2..2....2..211112.
211112...2....2...211112...2....2..211.12..2....2.
2.1112...211112...2.1112...211112..211112..2....2.
211112...2.1112...211112...211112..211.121.2....2.
2.1112...211112...2.1112...211112..211.12..2111.2.
2....2...211112...2....2...2.1112..2.1112..2.1112.
2....2...211.12...2....2...211112..2....2..2111.2.
2....2...2....2...2....2...2....2..2....2..211112.
211112...2....2...211112...2....2..2....2..2.1112.
211112...2....21..2111.2...2....2..21.112..211112.
2.1112...2..112...2.1112...211112..211112..2....2.
211112...211112...211112...2.1112..2.1112..2....2.
2.1112...2111.2...211112.1.211112..211112..2....2.
211112..1211112...2.1112...2.1112..211112..2.1112.
```

```
2 . . . . 2 . . . 2 . 1 1 1 2 . . . 2 . . . . 2 . . . 2 1 1 1 1 2 . . 2 1 1 1 1 2 . . 2 . 1 1 1 2 .
2 . . . . 2 . . . 2 1 1 1 1 2 . . . 2 . . . . 2 . . . 2 . 1 1 1 2 . . 2 . . . . 2 . . 2 1 1 1 1 2 .
2 2 2 2 2 2 . . . 2 2 2 2 2 2 . . . 2 2 2 2 2 2 . . . 2 2 2 2 2 2 . . 2 2 2 2 2 2 . . 2 2 2 2 2 2 .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 1 1 1 1 .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

## outFile2 Zone2

```
Below is HPP
0 7 15 24 22 26 16 11 5 11 14 25 24 25 15 11 5 15 14 21 22 24 17 13 4 14 14 21 22 26 15 11 5 14 19 20 21 20 26 16 13 9 5 0 0 0 0
Below is VPP
0 16 26 26 26 22 0 1 1 0 17 22 25 25 18 1 0 1 1 20 26 26 22 14 0 1 0 1 20 27 27 24 19 0 0 0 19 22 23 26 18 1 0 0 19 26 26 23 19 0 0 0
Below is binHPP
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
Below is binVPP
0 1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 0 1 1 1 1 1 0 0 0 1 1 1 1 1 0 0 0
Below is the linked list after insert input zone box
Below is morphHPP, after performing morphCLosing on HPP
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
Below is morphVPP after performing morphClosing on VPP
0 1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 0 1 1 1 1 1 0 0 0 1 1 1 1 1 0 0 0
The number of runs in morphHPP-runsHPP is 2
The number of runs in morphVPP-runsVPP is 7
readingDirection is 2
2
1 44 42 49
2
1 36 42 41
2
1 28 42 33
2
1 19 42 24
2
1 10 42 15
2
1 1 42 6
```

## outFile1 Zone3

```
Below is the input image
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . .
. . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . 1 1 1 . . . . . . . .
. . . . . 1 1 1 1 1 . . . . . . . . . . . . . . . . . . 1 1 1 1 1 . . . . . . .
. . . . 1 1 1 1 1 1 . . . . . . . . . . . . . . . . . 1 1 1 1 1 1 1 . . . . . .
. . . . . 1 1 1 1 1 . . . . . . 1 . . . . . . . . . 1 1 1 1 1 1 1 1 1 . . . . .
. . . . . . . 1 1 1 . . . . . . 1 . . . . . . . . 1 1 1 1 1 1 1 1 1 1 1 . . . .
. . . . 1 1 1 1 1 1 . . . . . . 1 . . . . . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 . . .
. . . . 1 1 1 1 1 1 . . . . . 1 1 1 . . . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . .
. . . . 1 1 1 1 1 1 . . . . 1 1 1 1 1 . . . . . . . 1 1 1 1 1 1 1 1 1 . . . . .
. . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . . . . . . 1 1 1 1 1 1 1 . . . . . .
. . . . 1 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1 . . . . . . . . . 1 1 1 . . . . . . . .
. . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . .
. . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 . . .
. . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . . . . 1 1 1 1 1 1 1 1 1 1 1 . . . .
. . . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . . . 1 1 1 1 1 1 1 1 1 1 . . . . . .
. . . . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . 1 1 1 1 1 . . . . . . .
. . . . . . . 1 1 1 1 1 . . . 1 1 1 1 1 1 1 1 1 1 1 1 . 1 1 1 1 . . . . . . . .
. . . . . . . . 1 1 1 . . . . . 1 1 1 1 . . 1 1 1 . . . . 1 1 . . . . . . . . .
. . . . . . . . 1 1 . . . . . . . 1 1 . . . . 1 1 . . . . . 1 . . . . . . . . .
. . . . . . . . 1 1 . . . . . . . 1 1 . . . . 1 1 . . . . . 1 . . . . . . . . .
. . . . . . . . 1 1 . . . . . . . 1 1 . . . . 1 1 . . . . . 1 . . . . . . . . .

The zone may be a non-text zone
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . .
. . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . 1 1 1 . . . . . . . .
. . . . . 1 1 1 1 1 . . . . . . . . . . . . . . . . . . 1 1 1 1 1 . . . . . . .
. . . . 1 1 1 1 1 1 . . . . . . . . . . . . . . . . . 1 1 1 1 1 1 1 . . . . . .
. . . . . 1 1 1 1 1 . . . . . . 1 . . . . . . . . . 1 1 1 1 1 1 1 1 1 . . . . .
. . . . . . . 1 1 1 . . . . . . 1 . . . . . . . . 1 1 1 1 1 1 1 1 1 1 1 . . . .
. . . . 1 1 1 1 1 1 . . . . . . 1 . . . . . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 . . .
. . . . 1 1 1 1 1 1 . . . . . 1 1 1 . . . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . .
. . . . 1 1 1 1 1 1 . . . . 1 1 1 1 1 . . . . . . . 1 1 1 1 1 1 1 1 1 . . . . .
. . . . 1 1 1 1 1 1 . . . 1 1 1 1 1 1 . . . . . . . . 1 1 1 1 1 1 1 . . . . . .
. . . . 1 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1 . . . . . . . . . 1 1 1 . . . . . . . .
. . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . .
. . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 . . .
. . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . . . . 1 1 1 1 1 1 1 1 1 1 1 . . . .
. . . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . . . 1 1 1 1 1 1 1 1 1 1 1 . . . . . .
. . . . . . 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 . 1 1 1 1 1 1 . . . . . . .
. . . . . . . 1 1 1 1 1 . . . 1 1 1 1 1 1 1 1 1 1 1 1 . 1 1 1 1 1 . . . . . . . .
. . . . . . . . 1 1 1 . . . . . 1 1 1 1 . . 1 1 1 . . . . 1 1 . . . . . . . . .
. . . . . . . . 1 1 . . . . . . . 1 1 . . . . 1 1 . . . . . 1 . . . . . . . . .
. . . . . . . . 1 1 . . . . . . . 1 1 . . . . 1 1 . . . . . 1 . . . . . . . . .
. . . . . . . . 1 1 . . . . . . . 1 1 . . . . 1 1 . . . . . 1 . . . . . . . . .
```

## outFile2 Zone3

```
Below is HPP
0 2 4 10 13 15 15 20 24 20 19 18 32 30 28 26 26 21 12 7 7 7 0 0
Below is VPP
0 0 0 0 9 12 13 17 19 19 8 6 6 7 8 10 14 14 13 8 6 2 3 8 11 9 11 11 14 17 21 16 13 11 8 6 4 2 0 0 0 0
Below is binHPP
0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
Below is binVPP
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0
Below is the linked list after insert input zone box
Below is morphHPP, after performing morphCLosing on HPP
0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
Below is morphVPP after performing morphClosing on VPP
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0
The number of runs in morphHPP-runsHPP is 2
The number of runs in morphVPP-runsVPP is 2
readingDirection is -437714681
```