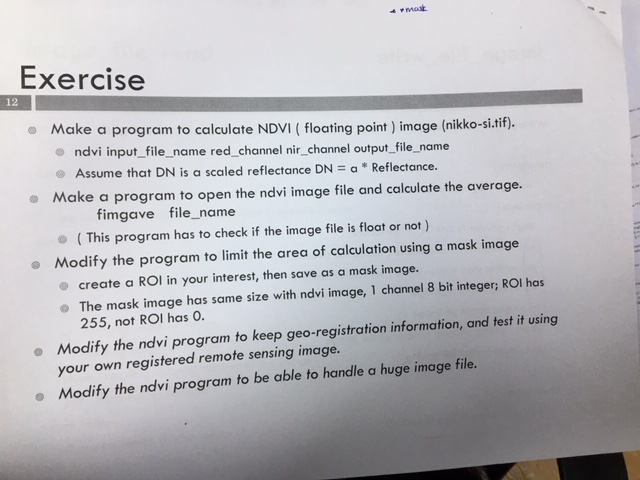
**Handling float/short image**

**Objective**

A program to calculate NDVI (floating point) image (nikko-si.tif)



**Code Descriptions**

//declare directives//

#include<stdio.h>

#include<math.h>

#include "image.h"

#include "imgio.h"

void usage(void)

{

fprintf(stderr,"usage: ndvi inputfile redBandIndex NIRBandIndex outputFile\n");

}

//DECLARE FUNCTIONS//

//function declaration for NDVI calculation.(float \*\*pixO) to access to each pixel of output image, a pointer of float data type prepared//

int ndvi\_cal(u\_char \*\*vr, u\_char \*\*nir, float \*\*pixO, int w, int h);

//function declaration for NDVI mask calculation//

int ndvi\_mask\_cal(u\_char \*\*red, u\_char \*\*nir, float \*\*pixO, u\_char \*\*mask, int w, int h);

//main part of function//

//define variables; argc, argv, \*imgI, \*imgO, w,h,nchan, Vr\_Index, NIR\_Index, \*imageFnameI, \*imageFnameO,\*\*vr, \*\*nir //

void main(int argc, char \*\*argv[]){

IMAGE \*imgI,\*imgO; // Image input, Image output

IMAGE\_FILE \*imgfI, \*imgfO; // ImageFile input, ImageFile Output

int w, h, nchan,vr\_index, nir\_index; // width, height, VR index, NIR index

char \*imageFnameI, \*imageFnameO; // Image Filename Input, Image Filename Output

u\_char \*\*vr, \*\*nir;

if (argc !=5){ // check amount of input arguments

usage();

exit(1);

}

// assign arguments value to variable

/\*receive image file, number of NIR band, number of Red band and output image file name from user\*/

imageFnameI = argv[1];

vr\_index = atoi(argv[2]);

nir\_index = atoi(argv[3]);

imageFnameO = argv[4];

// image file open input image  
if ((imgfI = image\_file\_open(imageFnameI, IMAGE\_RDONLY, IMAGE\_BUFFERED)) == NULL){ //if error in image file name, read only or buffered ; exit the program//

printf("Image\_file\_open\_error\n");

exit(2);

}

//obtain spec of the image without data memory//

imgI = imgfI->image;

w = imgI->w;

h = imgI->h;

vr = imgI->data[vr\_index - 1]; // get Red Image

nir = imgI->data[nir\_index - 1]; // get NIR Image

imgO = image\_alloc(w, h, IMAGE\_FLOAT,1); //image allocation with image\_float and band number 1 is going to use

if (imgO == NULL){

fprintf(stderr, "contrast: output image file allocate error %d %d %d\n", w, h, nchan);

exit(4);

}

//Calling NDVI calculation function

ndvi\_cal(vr, nir, imgO->data[0], w, h);

//create a new file output//

imgfO = image\_file\_create(imageFnameO, IMAGE\_TRUNC, 0, imgO);

if (imgfO == NULL){

fprintf(stderr, "contrast: output image file create error %s\n", imageFnameO);

exit(5);

}

image\_file\_close(imgfO); // close output file

image\_destroy(imgO); // destroy image

image\_file\_close(imgfI); // close input file

return 0;

}

/\*Function to calculate NDVI image pixel by pixel\*/

int ndvi\_cal(u\_char \*\*vr, u\_char \*\*nir, float \*\*pixO, int w, int h){

int i, j;

for (i = 0; i < h; i++){

for (j = 0; j < w; j++){

pixO[i][j] = (float)(((double)nir[i][j] - (double)vr[i][j]) / ((double)nir[i][j] + (double)vr[i][j]));

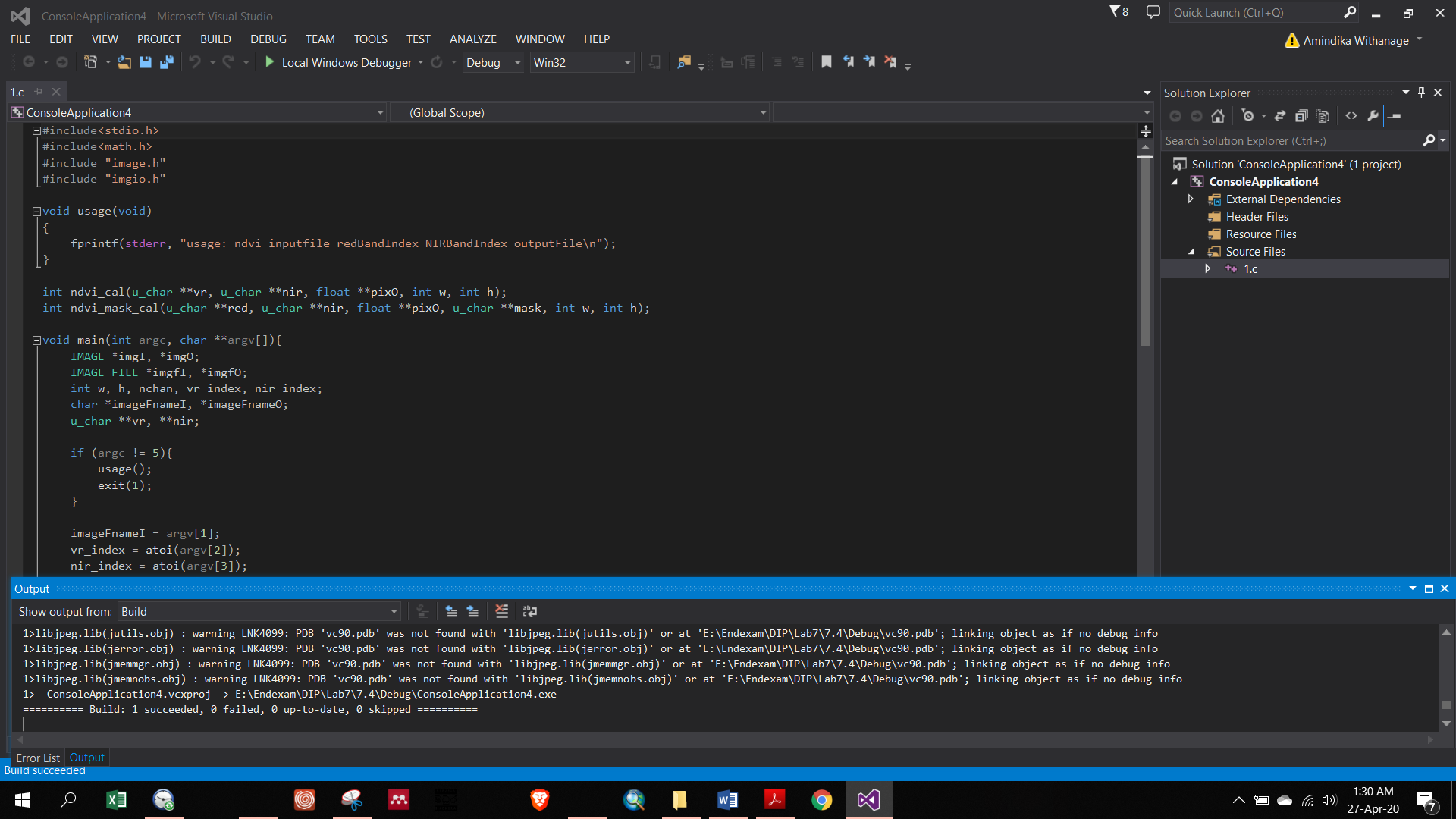
}

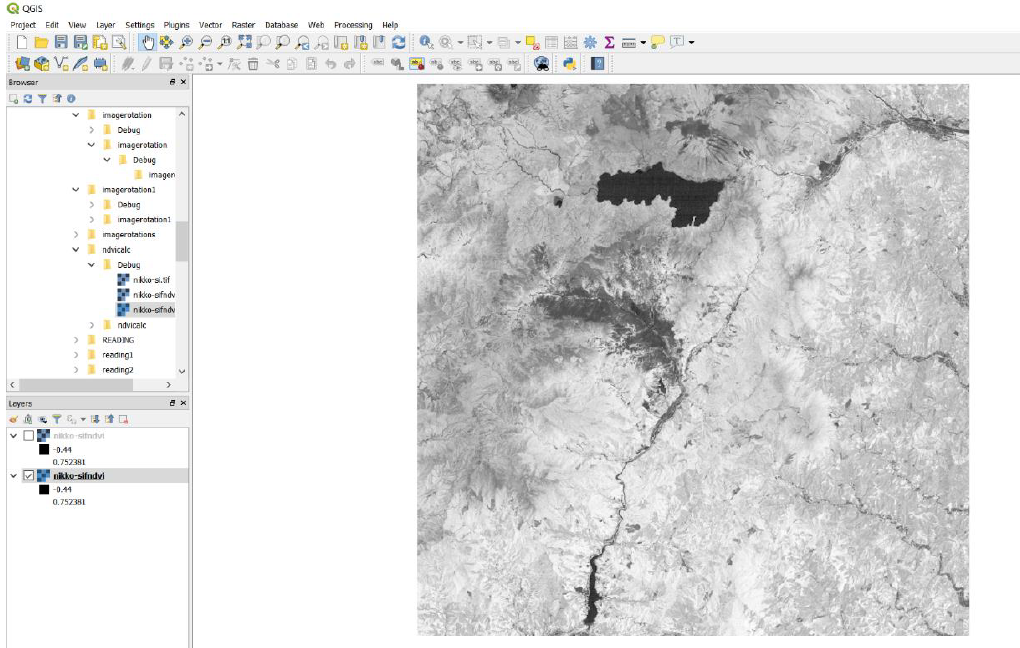
}

return 0;

}

**Output**





**Figure: NDVI image**