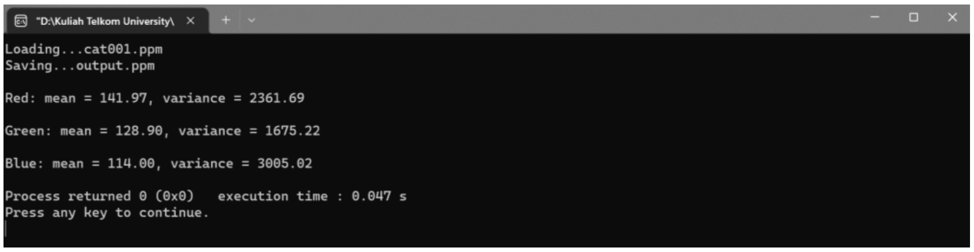
Name : Muhammad Daffa Khairi
ID : 2246176007

Exercise A01

- **Goal**

  Write a program to load the color image "cat001.ppm", and calculate **means** and **variances** of each signal in the image.

- **Output**



```
Loading...cat001.ppm
Saving...output.ppm

Red: mean = 141.97, variance = 2361.69

Green: mean = 128.90, variance = 1675.22

Blue: mean = 114.00, variance = 3005.02

Process returned 0 (0x0)   execution time : 0.047 s
Press any key to continue.
```

- **Code Specification**

Filename : **varmean.c**	
Specification of Function	
Name	find_mean
Arguments	Bitmap* bmp (Image data) float* result (Mean result)
Return Value	An array of float value
Summary	Means value calculation for each RGB channels
Specification of Function	
Name	find_variance
Arguments	Bitmap* bmp (Image data) float* result (Variance result) float* mean_result (Mean_result)
Return Value	An array of float value
Summary	Variances value calculation of each RGB channels

- **Code Algorithm**

Algorithm find_mean

Variables :	bmp	pointer of image data
	result	pointer of mean result
	total_arr	float
	red_sum	float
	green_sum	float
	blue_sum	float

```
Set {0, 0, 0} to result
For total_array i in the image data bmp
        red_sum ← red_sum + bmp->rmap[i]
        green_sum ← green_sum + bmp->gmap[i]
        blue_sum ← blue_sum + bmp->bmap[i]
result[0] = red_sum / total_array
result[1] = green_sum / total_array
result[2] = green_sum / total_array
Return result
```

Algorithm find_variance

Variables :	bmp	pointer of image data
	result	pointer of variance result
	mean_result	pointer of mean result
	total_arr	float
	red_Vsum	float
	green_Vsum	float
	blue_Vsum	float

```
Set {0, 0, 0} to result
For total_array i in the image data bmp
        red_Vsum ← red_Vsum + pow((bmp->rmap[i]- mean_result[0]),2)
        green_Vsum ← green_Vsum + pow((bmp->gmap[i]- mean_result[0]),2)
        blue_Vsum ← blue_Vsum + pow((bmp->bmap[i]- mean_result[0]),2)
result[0] = red_Vsum / total_array
result[1] = green_Vsum / total_array
result[2] = green_Vsum / total_array
Return result
```

- **Code**

1. main.c

```
#include "includes.h"
/**************************************************/
/* Main Function                          */
/**************************************************/
int main(int argc, char* argv[])
{
    Bitmap* inIM, * outIM;
    char* inName  = "cat001.ppm";
    char* outName = "output.ppm";
    int x, y;
    printf("Loading...%s\n", inName);
    inIM = loadPpm(inName);

    /* Definition of image structure for output */
    outIM = (Bitmap*)malloc(sizeof(Bitmap));
    if (outIM == NULL) {
            fprintf(stderr, "can't allocate memory.\n");
            exit(1);
    }
    outIM->format = inIM->format;
    outIM->width  = inIM->width;
    outIM->height = inIM->height;
    outIM->rmap = outIM->gmap = outIM->bmap = outIM->map = NULL;
    outIM->rmap = (unsigned char*)malloc(outIM->width * outIM->height * sizeof(unsigned char));
    outIM->gmap = (unsigned char*)malloc(outIM->width * outIM->height * sizeof(unsigned char));
    outIM->bmap = (unsigned char*)malloc(outIM->width * outIM->height * sizeof(unsigned char));
    if (outIM->rmap == NULL || outIM->gmap == NULL|| outIM->bmap == NULL) {
            fprintf(stderr, "can't allocate memory.\n");
            exit(1);
    }
    /* Copying */
    int p = 0;
    for (y = 0; y < inIM->height; y++) {
            for (x = 0; x < inIM->width; x++) {
                    outIM->rmap[y * outIM->width + x] = inIM->rmap[y * inIM->width + x];
                    outIM->gmap[y * outIM->width + x] = inIM->gmap[y * inIM->width + x];
                    outIM->bmap[y * outIM->width + x] = inIM->bmap[y * inIM->width + x];
      p++;
    }
    }
    printf("Saving...%s\n", outName);
    savePpm(outName, outIM);

  /* My Code */
  float mean_result[3] = {0,0,0}, variance_result[3] = {0,0,0};
  find_mean(outIM, mean_result);
  find_variance(outIM, variance_result, mean_result);
  printf("\nRed: mean = %.2f, variance = %.2f\n", mean_result[0], variance_result[0]);
  printf("\nGreen: mean = %.2f, variance = %.2f\n", mean_result[1], variance_result[1]);
  printf("\nBlue: mean = %.2f, variance = %.2f\n", mean_result[2], variance_result[2]);
    return 0;
}

/* Muhammad Daffa Khairi - 2246176007 */
```

2. includes.h

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <float.h>

#define PPM 6
#define PGM 5
#define PBM 4
#define BUFF 1024

#define RAISE(x) ((int)(x)+(((float)(x)==(int)(x))?0:1))

typedef struct _bitmap {
        char format;
        int width;
        int height;
        unsigned char* rmap;
        unsigned char* gmap;
        unsigned char* bmap;
        unsigned char* map;
} Bitmap;

#include "ppm.h"
#include "pgm.h"
#include "pbm.h"
#include "varmean.h“
```

3. varmean.c

```
#include "includes.h"
void find_mean(Bitmap* bmp, float * result){
   float total_arr = bmp->height * bmp->width, red_sum = 0, green_sum = 0, blue_sum = 0;
   for(int i = 0 ; i < total_arr;i++){
      red_sum = red_sum + bmp->rmap[i];
      green_sum = green_sum + bmp->gmap[i];
      blue_sum = blue_sum + bmp->bmap[i];
   }
   result[0] = red_sum / total_arr;
   result[1]= green_sum / total_arr;
   result[2] = blue_sum / total_arr;
}
void find_variance(Bitmap* bmp, float * result, float * mean_result){
   float total_arr = bmp->height * bmp->width, red_Vsum = 0, green_Vsum = 0, blue_Vsum = 0;
   for(int i = 0 ; i < total_arr;i++){
      red_Vsum = red_Vsum + pow((bmp->rmap[i]- mean_result[0]),2);
      green_Vsum = green_Vsum + pow((bmp->gmap[i]- mean_result[0]),2);
      blue_Vsum = blue_Vsum + pow((bmp->bmap[i]- mean_result[0]),2);
   }
   result[0] = red_Vsum / total_arr;
   result[1] = green_Vsum / total_arr;
   result[2] = blue_Vsum / total_arr;
}
```

4. varmean.h

```
/* varmean.h */

void mean(Bitmap* bmp, float * result);
void find_variance(Bitmap* bmp, float * result, float * mean_result)
```

5. Makefile

```
SHELL           = /bin/sh
CC              = gcc
CFLAGS                  = -g -Wall
LDFLAGS                 = -g -Wall -lm

#--------------- You must write all linked file name ----------------
SRCS   = main.c ppm.c pgm.c pbm.c varmean.c
#-------------------------------------------------------------------
OBJS    = ${SRCS:.c=.o}

#-------- Please write executive file name and directory ------------
DIR    = ./
BINS    = a01-01
#-------------------------------------------------------------------

all: ${BINS}

${BINS}: ${OBJS}
	${CC} -o ${DIR}$@ ${OBJS} ${LDFLAGS}

.c.o:
	${CC} ${CFLAGS} -c $<

clean:
	rm -f ${OBJS} core
```