

```
/******
```

```
WAVE HEIGHT CALCULATION VER 1.0 [wave_height_calc.cpp]
```

```
*****/
```

```
/*
```

```
Creator : Putu Hangga Nan Prayoga
```

```
Student ID : 2TE11450S
```

```
<concept of this program>
```

```
1)raw wave data exported from .txt to spreadsheet .xls and combined using func "X"&"&"Y"  
to create one row data, to be used in this program
```

```
2)Use floating point instead of integer to get precision
```

```
3)zero_up crossing point handled as location data instead of value
```

```
4)calculating algorithm is taken in main routine <function of this program>
```

```
*/
```

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
#include <time.h>
```

```
#include <math.h>
```

```
#define data 1024
```

```
float wave_data[data]=
```

```
{
```

```
-0.14,-0.1,-0.25,-0.39,-0.31,-0.18,-0.12,-0.15,-0.09,0.15,0.44,0.66,0.7,0.53,0.2,-0.09,-0.27,  
-0.32,-0.41,-0.48,-0.34,-0.16,0.11,0.1,0.05,0.13,0.23,0.38,0.48,0.33,0.11,-0.06,-0.16,-0.29,  
-0.31,-0.33,-0.42,-0.33,-0.14,-0.02,0.17,0.31,0.17,0.06,0.17,0.26,0.31,0.15,-0.09,-0.21,-0.24,  
-0.22,-0.17,-0.09,-0.07,-0.01,0.15,0.2,0.3,0.26,0.09,-0.03,0.02,-0.01,-0.25,-0.58,-0.55,-0.28,  
0.11,0.22,0.03,0.03,0.18,0.37,0.41,0.34,0.31,0.19,0.04,-0.02,0,-0.19,-0.41,-0.47,-0.45,-0.23,  
-0.01,-0.04,-0.12,-0.08,0.03,0.02,0.04,0.08,0.12,0.07,0.15,0.28,0.43,0.64,0.68,0.32,-0.19,  
-0.59,-0.79,-0.7,-0.4,-0.1,0.01,0.03,0.03,-0.05,0.08,0.14,0.11,0.17,0.24,0.31,0.31,0.32,0.08,  
-0.13,-0.2,-0.22,-0.21,-0.15,-0.13,-0.07,-0.04,-0.1,-0.13,-0.07,0.05,0.12,0.03,-0.3,-0.42,  
-0.28,0.06,0.49,0.72,0.72,0.54,0.34,0.2,-0.06,-0.35,-0.53,-0.67,-0.75,-0.53,-0.24,-0.02,0.24,  
0.39,0.32,0.28,0.24,0.18,-0.07,-0.35,-0.2,0.12,0.36,0.39,0.27,0.15,-0.06,-0.27,-0.27,-0.19,  
-0.09,-0.04,-0.15,-0.1,-0.24,-0.3,-0.09,0.05,-0.01,-0.07,0.07,0.33,0.47,0.37,0.19,0.01,-0.12,  
-0.23,-0.41,-0.45,-0.17,0.22,0.38,0.51,0.78,0.61,0.13,-0.27,-0.51,-0.65,-0.57,-0.36,-0.17,  
-0.01,0.01,0,-0.01,-0.15,-0.16,-0.03,0.11,0.25,0.33,0.25,0.2,0.18,0.27,0.33,0.34,0.29,0.09,  
-0.24,-0.6,-0.9,-0.93,-0.69,-0.2,0.21,0.52,0.77,0.72,0.52,0.25,0,-0.08,-0.14,-0.16,-0.12,  
-0.06,0.01,0.09,0.12,-0.2,-0.57,-0.68,-0.64,-0.36,-0.04,0.14,0.17,0.21,0.42,0.55,0.64,0.74,  
0.53,0.13,-0.27,-0.56,-0.47,-0.18,0.13,0.15,0.15,0.2,0.14,-0.23,-0.54,-0.63,-0.62,-0.43,  
-0.02,0.17,0.16,0.11,0.15,0.11,0.04,0.05,0.17,0.31,0.46,0.54,0.54,0.41,0.18,-0.14,-0.38,-0.33,  
-0.23,-0.2,-0.22,-0.32,-0.44,-0.56,-0.55,-0.4,-0.29,-0.05,0.13,0.34,0.61,0.67,0.6,0.48,0.11,  
-0.12,-0.04,0.12,0.21,0.32,0.37,0.31,0.14,0.1,-0.01,-0.34,-0.71,-0.96,-1.16,-1.19,-0.92,-0.53,  
-0.14,0.15,0.4,0.69,1.04,1.16,1.08,0.82,0.5,0.37,0.22,0.07,-0.05,-0.35,-0.52,-0.52,-0.48,-0.34,  
-0.27,-0.42,-0.59,-0.65,-0.51,-0.23,0.12,0.35,0.42,0.33,0.25,0.1,0.2,0.27,0.22,0.16,0.12,0.18,
```

0.27,0.34,0.2,0.01,-0.09,-0.21,-0.39,-0.41,-0.38,-0.34,-0.2,0.14,0.37,0.19,-0.07,-0.23,-0.16,
-0.04,0.01,0.07,0.15,0.18,0.09,-0.07,0,0.04,0.01,-0.11,-0.25,-0.26,-0.08,0.01,0.09,0.2,0.29,
0.44,0.48,0.47,0.38,0.15,-0.16,-0.32,-0.47,-0.51,-0.48,-0.29,-0.22,-0.22,-0.17,-0.23,-0.27,
-0.25,-0.28,-0.09,0.31,0.6,0.83,0.87,0.68,0.27,0.01,0.08,0.15,0.14,0.01,-0.23,-0.28,-0.45,-0.58,
-0.5,-0.29,-0.16,-0.06,-0.06,-0.16,-0.19,-0.11,-0.06,0.01,0.05,0.08,0.18,0.27,0.38,0.54,0.54,
0.4,0.09,-0.44,-0.65,-0.41,0.06,0.41,0.59,0.47,0.2,0.03,-0.12,-0.24,-0.36,-0.52,-0.54,-0.52,-0.5,
-0.37,-0.19,0.1,0.12,0.09,0.22,0.4,0.56,0.51,0.27,0.04,0.06,0.21,0.36,0.26,0.1,-0.05,-0.04,-0.02,
-0.18,-0.47,-0.71,-0.71,-0.61,-0.24,0.07,0.15,0.1,0.13,0.17,0.25,0.31,0.42,0.46,0.45,0.18,-0.23,
-0.66,-0.64,-0.2,0.29,0.64,0.69,0.52,0.1,-0.2,-0.34,-0.34,-0.34,-0.4,-0.3,-0.16,-0.18,-0.36,-0.46,
-0.5,-0.38,-0.04,0.43,0.71,0.62,0.51,0.47,0.43,0.33,0.28,0.3,0.21,0.12,0.06,-0.16,-0.34,-0.51,
-0.55,-0.64,-0.76,-0.63,-0.29,0.04,0.28,0.34,0.23,0.17,0.04,0.1,0.24,0.17,-0.02,0.08,0.22,0.38,
0.5,0.37,0.32,0.09,-0.2,-0.42,-0.55,-0.48,-0.37,-0.36,-0.33,-0.26,-0.18,-0.01,0.09,0.13,0.15,
0.19,0.34,0.37,0.44,0.48,0.44,0.44,0.39,0.1,-0.25,-0.48,-0.49,-0.46,-0.46,-0.4,-0.4,-0.41,-0.33,
-0.31,-0.14,0.08,0.07,0.13,0.25,0.41,0.51,0.56,0.65,0.69,0.5,0.2,-0.09,-0.3,-0.38,-0.3,-0.02,
0.16,0.12,-0.23,-0.63,-0.88,-1.01,-0.7,-0.35,0.02,0.4,0.53,0.46,0.41,0.38,0.37,0.4,0.21,0.01,
-0.07,-0.03,0.06,-0.05,-0.17,-0.1,0.14,0.36,0.41,0.36,0.04,-0.31,-0.56,-0.6,-0.62,-0.61,-0.41,
-0.2,0.01,0.13,0.11,0.05,0.08,0.39,0.76,0.73,0.34,0.01,-0.03,-0.16,-0.28,-0.32,-0.11,0.2,0.14,
0.08,0.15,0.13,0.09,0.03,-0.03,-0.01,-0.13,-0.36,-0.53,-0.63,-0.69,-0.44,0.09,0.6,0.66,0.56,
0.46,0.26,0.09,-0.01,-0.14,-0.14,-0.21,-0.39,-0.36,-0.15,-0.04,0.16,0.35,0.48,0.48,0.35,0.04,
-0.25,-0.51,-0.64,-0.5,-0.09,0.25,0.36,0.46,0.48,0.29,0.04,-0.25,-0.39,-0.34,-0.16,-0.11,-0.12,
-0.1,-0.05,-0.03,-0.07,-0.16,-0.22,-0.24,-0.24,-0.02,0.37,0.77,0.99,0.88,0.47,-0.01,-0.31,-0.4,
-0.5,-0.43,-0.14,0.02,0.01,-0.02,-0.08,-0.19,-0.23,-0.26,-0.37,-0.33,-0.09,0.23,0.44,0.36,0.11,
-0.11,-0.03,-0.03,-0.11,-0.01,0.2,0.49,0.47,0.37,0.25,0.1,0.02,-0.18,-0.28,-0.23,-0.18,-0.17,
-0.27,-0.41,-0.53,-0.47,-0.35,-0.21,0.01,0.29,0.54,0.64,0.56,0.37,0.08,-0.23,-0.5,-0.38,-0.1,
0.2,0.48,0.65,0.62,0.43,0.23,-0.13,-0.46,-0.74,-0.68,-0.45,-0.13,0.08,0,-0.03,-0.01,-0.06,
-0.35,-0.35,-0.19,-0.08,0.03,0.25,0.37,0.4,0.38,0.21,0.17,0.19,0.3,0.34,0.28,0.11,0,-0.15,-0.39,
-0.42,-0.12,0.02,-0.17,-0.36,-0.38,-0.3,-0.16,0.19,0.4,0.23,0.17,-0.04,-0.28,-0.51,-0.35,0.05,
0.3,0.29,-0.01,-0.13,-0.01,0.35,0.65,0.59,0.29,-0.01,-0.25,-0.41,-0.47,-0.28,-0.05,0.09,0.14,
0.18,0.21,0.05,-0.17,-0.09,-0.03,-0.09,-0.02,0.1,0,-0.03,-0.08,-0.1,-0.2,-0.29,-0.06,0.21,0.14,
0,0,-0.01,-0.16,-0.16,-0.05,0.16,0.13,-0.23,-0.37,-0.09,0.46,0.7,0.69,0.58,0.35,0,-0.21,-0.08,
-0.01,-0.04,-0.07,-0.29,-0.55,-0.68,-0.73,-0.5,-0.09,0.31,0.42,0.24,-0.24,-0.35,-0.19,-0.08,
0.01,0.18,0.33,0.33,0.34,0.4,0.49,0.65,0.6,0.37,-0.14,-0.46,-0.33,-0.12,-0.12,-0.21,-0.5,-0.6,
-0.25,-0.05,-0.21,-0.3,-0.32,-0.3,-0.21,-0.02,0.21,0.51,0.82,0.97,0.74,0.34,-0.02,-0.4,-0.57,
-0.54,-0.44,-0.31,-0.21,0.06,0.41,0.52,0.19,-0.18,-0.26,-0.24,-0.11,-0.08,-0.1,-0.01,0.16,0.36,
0.6,0.6,0.38,0.08,-0.26,-0.48,-0.52,-0.37,-0.35,-0.39,-0.29,-0.15,-0.03,0.01,0.19,0.4,0.56,
0.53,0.36,0.02,-0.26,-0.41,-0.47,-0.4,-0.17,0.11,0.19,0.05,0.17,0.4,0.38,0.35,0.2,0.07,0.05,
-0.01,-0.19,-0.22,-0.25,-0.3,-0.19,0,0.06,0.01,-0.12,-0.2,-0.27,-0.4,-0.49,-0.4,-0.02,0.27,
0.5,0.59,0.62,0.57,0.37,-0.07,-0.33,-0.23,-0.03,0.14,0.27,0.25,0.17,-0.11,-0.24,-0.28,-0.28

```

};
int a,b,x,y;
float min[100];
float max[100];
float corr;
float fdata;
float sum_data;
float dum_wave_data[data];
float wave_data2[data];
float dum_wave[data];
float zero_up[100];
int num [100];
float wave_height[100];
float wave_period[100];

// function declaration
void mean_correction (void);
void search_max_min (void);
void zero_up_point (void);
void sorting (void);
void print_data (void);

FILE *fp; //file open and close

FILE *file_open(char *file_name,char *mode)
{
    FILE *fp;
    if ((fp=fopen(file_name,mode))==NULL) {
        fprintf (stderr,"cannot open a file\n");
        exit(1);
    }
    return fp;
}

void mean_correction (void)
{
// change the value of data number from int to floating point
    fdata=data;
    for (a=0;a<data;a++){
        sum_data+=wave_data[a];
    }

//correction of mean water level//
    corr = sum_data/fdata;
//get new wave data from the mean water level correction//
    for (a=0;a<data;a++){

```

```

        dum_wave_data[a]=wave_data[a]-corr;
        wave_data2[a]=dum_wave_data[a];
    }
}

void zero_up_point (void)
{
    float wave_check[data];
    float wave_front[data];

//determining criteria for zero-upcrossing point//
//defining criteria of ni*ni+1<0//
    for (a=0;a<data-1;a++){
        wave_check[a] = wave_data2[a]*wave_data2[a+1];
    }
//defining criteria of ni+1>0
    for (a=0;a<data-1;a++){
        wave_front[a] = wave_data2[a+1];
    }

//detecting the location of zero upcrossing point//
// x is the var for zero-up value, y is the var for zero-up location in the data//
//if according to criteria, save the value and location into new variable wave_data3 & num//
    x=1;
    y=1;
    for (a=1;a<data;a++){
        if (wave_check[a]<0 & wave_front[a]>0){
            dum_wave[x]= wave_data2[a];
            x++;
            num[y]=a;
            y++;
        }
    }
}

void search_max_min(void)
{
//now we use the location (num) to determine the boundary for maxima & minima searching algorithm//
//first, set the initial value of max & min of each wave profile//
//then decide looping criteria for each wave profile, boundary is between 2 crossing point.//
//The last is calculation of H & T for each wave profile//

//looping process for taking max & min value for each wave profile//
    for (a=1;a<x-1;a++){
        max [a]=0;
        min [a]=0;
    }
}

```

```

        for (b=num[a];b<num[a+1];b++){
            if (wave_data2[b]>max[a])
                max[a]=wave_data2[b];

            if (wave_data2[b]<min[a])
                min[a]=wave_data2[b];
        }
        wave_height[a]=max[a]-min[a];
        wave_period[a]=(num[a+1]-num[a])*0.5;
    }
}

void sorting (void)
{
    int i_i,i_j;
    float i_temp,j_temp;
    float temp[100],temp2[100];
//Sorting of wave height (H) & period(T) in the descending order of H
    i_i=1;
    i_j=1;
    for (a=1;a<x-1;a++){
        temp[i_i] = wave_height[a];
        i_i++;
    }
    for (a=1;a<x-1;a++){
        temp2[i_j] = wave_period[a];
        i_j++;
    }

    for (a=1;a<x-1;a++){
        for (b=1;b<x-1;b++){
            if (temp[b]<temp[a]){
                i_temp = temp[b];
                temp[b] = temp[a];
                temp[a] = i_temp;
            }
            if (temp2[b]<temp2[a]){
                j_temp = temp2[b];
                temp2[b] = temp2[a];
                temp2[a] = j_temp;
            }
        }
    }

    fprintf (fp,"\n### Sorting of wave height (H) & period(T) in the descending order of H ###\n");
    fprintf (fp,"[No]= H   ,   T\n",a,temp[a],temp2[a]);
    for (a=1;a<x-1;a++){

```

```

        fprintf (fp,"[%d]= %f , %f\n",a,temp[a],temp2[a]);
    }
}

void print_data (void)
{
    fprintf (fp,"\nsum of wave_data= %f\n",sum_data);
    fprintf (fp,"\ncorrection of mean water level= %f\n",corr);
    fprintf (fp,"\n### new wave data after correction located in wave_data2[%d] ###\n",data);
//printing new wave data after correction
    /*for (a=0;a<data;a++){
        fprintf (fp,"\ncorr_wave_data [%d]= %f\n",a,wave_data2[a]);
    }*/
//printing the value of zero-up crossing point and location in the data //
    fprintf (fp,"\n### value of zero-up crossing point and location in the data### \n");
    for (a=1;a<x;a++){
        fprintf (fp,"wave_data2[%d]= %f\n",num[a],dum_wave[a]);
    }
//printing maxima & minima of each wave profile//
    fprintf (fp,"\n### maxima & minima point of each wave profile ###\n");
    for (a=1;a<x-1;a++){
        fprintf (fp,"from wave_data2[%d] - wave_data2[%d], max= %f ,
min= %f\n",num[a],num[a+1],max[a],min[a]);
    }
//printing output of wave height & wave period//
    fprintf (fp,"\n### Output of wave height & wave period ###\n");
    for (a=1;a<x-1;a++){
        fprintf (fp,"wave_height[%d]= %f",a,wave_height[a]);
        fprintf (fp," , wave_period[%d]= %f\n",a,wave_period[a]);
    }
}

void main(void)
{
    fp=file_open( "fp_wave_height_calculation.txt","w");
    fprintf (fp,"\n##### wave_height_calc_ver 1.0 #####\n");

    mean_correction ();
    zero_up_point ();
    search_max_min ();
    print_data ();
    sorting ();

    fprintf (fp,"\n##### END OF PROGRAM #####\n");
}

```