

Ahmad Hijazi

300047946

Assignment 5

Question 2

Code source:

```
// Author: Ahmad Hijazi
```

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

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

```
#define N 25 // display 25 values
```

```
#define ROW_IX 0
```

```
#define COLUMN_IX 1 // 1 column
```

```
typedef struct // structure defining
```

```
{
```

```
    char name[100];
```

```
    double n, slope, width, max_Depth;
```

```
}CHANNEL;
```

```
void getPositiveValue(CHANNEL);
```

```
void displayTable(CHANNEL);
```

```
double computeVelocity(double, CHANNEL);
```

```
int main(void)
```

```
{
```

CHANNEL components;

```
    getPositiveValue(components);
}

void getPositiveValue(CHANNEL identity)
{
    do{
        fflush(stdin);

        printf("Enter name of the channel:");

        gets(identity.name);

        printf("Enter coefficient of roughness:");

        scanf("%lf", &identity.n);

        printf("Enter the slope:");

        scanf("%lf", &identity.slope);

        printf("Enter channel width:");

        scanf("%lf", &identity.width);

        printf("Enter maximum depth of the channel:");

        scanf("%lf", &identity.max_Depth);

    }

    while ((identity.n<=0.0) || (identity.slope<=0.0) || (identity.width<=0.0) || (identity.max_Depth<=0.0));

    displayTable(identity);
}

void displayTable(CHANNEL ID )

{
    double points[2][N];

    double depth, inc;
```

```

int ix, jx;

inc= ID.max_Depth/N;

depth= ID.max_Depth/N;

ix=0;

printf("\n");

while(ix<N)
{
points[ROW_IX][ix]= depth;
points[COLUMN_IX][ix]= computeVelocity(depth, ID);
depth += inc;
ix++;
}

printf("Channel data for \"%s\" \n", ID.name);
printf("Coefficient of roughness: %f\n", ID.n);
printf("Slope %f\n", ID.slope);
printf("Width %f\n", ID.width);
printf("Maximum depth %f\n", ID.max_Depth);
printf("Depth   Average velocity\n");
printf("-----\n");
for(jx=0; jx<N; jx++)
printf("%.2f   %.4f\n", points[ROW_IX][jx], points[COLUMN_IX][jx]);
}

```

```

double computeVelocity(double height, CHANNEL features)

```

```

{
double v;
v= (features.width/height);

```

```

v= (v/(features.width + (2*height)));

v= pow(v,0.667);

v= v * sqrt(features.slope)/features.n;

return(v);

}

```

Code output:

```

Enter name of the channel:1
Enter coefficient of roughness:0.035
Enter the slope:0.0001
Enter channel width:10
Enter maximum depth of the channel:4.2

Channel data for "1"
Coefficient of roughness: 0.035000
Slope 0.000100
Width 10.000000
Maximum depth 4.200000
Depth      Average velocity
-----
0.17       0.9185
0.34       0.5663
0.50       0.4232
0.67       0.3424
0.84       0.2894
1.01       0.2514
1.18       0.2227
1.34       0.2001
1.51       0.1818
1.68       0.1666
1.85       0.1538
2.02       0.1428
2.18       0.1332
2.35       0.1249
2.52       0.1175
2.69       0.1109
2.86       0.1050
3.02       0.0996
3.19       0.0948
3.36       0.0904
3.53       0.0863
3.70       0.0826
3.86       0.0792
4.03       0.0760
4.20       0.0730

Process returned 0 (0x0)   execution time : 44.179 s
Press any key to continue.

```

```
Enter name of the channel:2
Enter coefficient of roughness:0.0013
Enter the slope:0.0032
Enter channel width:2
Enter maximum depth of the channel:11.5
```

```
Channel data for "2"
Coefficient of roughness: 0.001300
Slope 0.003200
Width 2.000000
Maximum depth 11.500000
```

```
Depth      Average velocity
```

```
-----
0.46       56.7478
0.92       29.7730
1.38       19.6859
1.84       14.4422
2.30       11.2593
2.76       9.1389
3.22       7.6350
3.68       6.5186
4.14       5.6608
4.60       4.9835
5.06       4.4366
5.52       3.9871
5.98       3.6118
6.44       3.2943
6.90       3.0227
7.36       2.7881
7.82       2.5837
8.28       2.4041
8.74       2.2453
9.20       2.1040
9.66       1.9776
10.12      1.8639
10.58      1.7612
11.04      1.6680
11.50      1.5831
```

```
Process returned 0 (0x0)   execution time : 42.936 s
Press any key to continue.
```

```

Enter name of the channel:3
Enter coefficient of roughness:0.17
Enter the slope:0.041
Enter channel width:40
Enter maximum depth of the channel:1.5

Channel data for "3"
Coefficient of roughness: 0.170000
Slope 0.041000
Width 40.000000
Maximum depth 1.500000
Depth      Average velocity
-----
0.06       7.7633
0.12       4.8797
0.18       3.7160
0.24       3.0612
0.30       2.6326
0.36       2.3266
0.42       2.0952
0.48       1.9129
0.54       1.7649
0.60       1.6419
0.66       1.5378
0.72       1.4483
0.78       1.3704
0.84       1.3018
0.90       1.2408
0.96       1.1863
1.02       1.1371
1.08       1.0925
1.14       1.0518
1.20       1.0145
1.26       0.9802
1.32       0.9484
1.38       0.9190
1.44       0.8916
1.50       0.8660

Process returned 0 (0x0)   execution time : 64.326 s
Press any key to continue.

```

Question 3:

Code source:

//Author Ahmad Hijazi

#include <stdio.h>

#include <math.h>

```

typedef struct
{

    char name[60];

    double area;

    double left_min, left_max, regrowth;

} FOREST;

double getPositiveValue(char *prompt);

void main()
{
    FOREST forest;
    double regrowth[20];
    int time[20];
    getForestInput(&forest);
    calculateTableData(&forest, regrowth, 20, time);
    displayTreeTable(regrowth, time, 20);
}

void getForestInput(FOREST *x)
{
    printf("Forest name:"); // enter 1 2 or 3
    fgets(x->name, 60, stdin);

    x->area=getPositiveValue("Total Acres:");

```

```

do
{
    x->left_min = getPositiveValue("Acres Uncut Minimum:");
}
while(x->left_min >= x->area);

do
{
    x->left_max = getPositiveValue("Acres Uncut Maximum:");
}
while(x->left_max <= x->left_min || x->left_max >= x->area);

do
{
    x->regrowth=getPositiveValue("Reforestation Rate:");
}
while(x->regrowth >= 1);
}

void calculateTableData(FOREST *y, double regrowth[], int h, int time[])
{
    double increment = (y->left_max-y->left_min)/(h-1);
    regrowth[0] = y->left_min;
    for(int i = 0; i < h; i++)
    {
        double new_area = regrowth[i];
        int j = 0;
        while(new_area < y->area)
        {

```



```

        new_area = new_area + y->regrowth*new_area;

        j++;
    }
    time[i] = j;
    regrowth[i+1] = regrowth[i]+increment;
}
}

```

```

void displayTreeTable(double r[], int t[], int h)
{
    printf("Uncut Area Start    # Years For Total Reforestation");
    for (int i = 0; i < h; i++)
        printf("\n %.2f        %d", r[i], t[i]);
}

```

```

double getPositiveValue(char *prompt)
{
    double value;
    do
    {
        printf(prompt);
        scanf("%lf",&value);
        if(value <= 0.0)
            printf("The value must be greater than zero.\n");
    }
    while(value <= 0.0);
    return (value);
}

```

Code output:

```
Forest name:1
Total Acres:15000
Acres Uncut Minimum:300
Acres Uncut Maximum:5000
Reforestation Rate:0.05
Uncut Area Start      # Years For Total Reforestation
300.00                81
547.37                68
794.74                61
1042.11              55
1289.47              51
1536.84              47
1784.21              44
2031.58              41
2278.95              39
2526.32              37
2773.68              35
3021.05              33
3268.42              32
3515.79              30
3763.16              29
4010.53              28
4257.89              26
4505.26              25
4752.63              24
5000.00              23
Process returned 20 (0x14)   execution time : 50.091 s
Press any key to continue.
```

```
Forest name:2
Total Acres:22676
Acres Uncut Minimum:1000
Acres Uncut Maximum:10000
Reforestation Rate:0.1
Uncut Area Start      # Years For Total Reforestation
1000.00              33
1473.68              29
1947.37              26
2421.05              24
2894.74              22
3368.42              21
3842.11              19
4315.79              18
4789.47              17
5263.16              16
5736.84              15
6210.53              14
6684.21              13
7157.89              13
7631.58              12
8105.26              11
8578.95              11
9052.63              10
9526.32              10
10000.00              9
Process returned 20 (0x14)   execution time : 37.370 s
Press any key to continue.
```

```
Forest name:3
Total Acres:58732
Acres Uncut Minimum:5000
Acres Uncut Maximum:20000
Reforestation Rate:0.15
Uncut Area Start      # Years For Total Reforestation
5000.00              18
5789.47              17
6578.95              16
7368.42              15
8157.89              15
8947.37              14
9736.84              13
10526.32             13
11315.79             12
12105.26             12
12894.74             11
13684.21             11
14473.68             11
15263.16             10
16052.63             10
16842.11              9
17631.58              9
18421.05              9
19210.53              8
20000.00              8
Process returned 20 (0x14)   execution time : 38.054 s
Press any key to continue.
```

Code Memory

```
#include <stdio.h>
#include <string.h>
#define SIZE_ARRAY_STR 15
// Prototypes
void appendString(char [], char []);
void main()
{
    char str1[SIZE_ARRAY_STR];
    char str2[SIZE_ARRAY_STR];
    // Initiates the character arrays
    // using standard functions
    strcpy(str1, "The ");
    strcpy(str2, "string");
    // Appends the contents of the array
    // str2 to the end of the string
    // in the array str1
    appendString(str1, str2);
}
/*-----
Description: The function appendsString appends the
            string referenced by s2 to the end
            of the string referenced by s1
            to the end of the string
            -----*/
void appendString(char s1[], char s2[])
{
    int ix1, ix2; // indexes of arrays
    // Finds the end of string 1
    // At the end of the loop,
    // s1[ix1] contains '\0'
    ix1 = 0;
    while(s1[ix1] != '\0') ix1 = ix1 + 1;
    // Now traverse string 2
    // to copy it into string 1
    for(ix2 = 0; s2[ix2] != '\0'; ix2 = ix2 + 1)
    {
        s1[ix1] = s2[ix2];
        ix1 = ix1 + 1;
    }
    s1[ix1] = '\0'; // to terminate the string
}
```

Working Memory

Str1

str2

?'T'	?'s'
?'h'	?'t'
?'e'	?'r'
?' 's'	?'l'
?' \0' 't'	?'n'
?'r'	?'g'
?'i'	?' \0'
?'n'	?
?'g'	?
?' \0'	?
?	?
?	?
?	?
?	?
?	?

S1 adr1

S2 adr2

ix1 ? 0 1 2 3 4 5 6 7 8 9

ix2 ? 0 1 2 3 4 5

UCT