VG101 Lab Worksheet

Lab 6

Instructor: Dr. Yifei ZHU
TA: Hangrui CAO
TA: Haoxuan SHAN
TA: Qinhang WU
TA: Muchen XU

Code Quality Requirement

- Ensure proper indentation
- Make naming meaningful
- Include necessary comments
- Split the code over functions
- Test the code as much as you can

Basic Syntax Exercises
- Function
  - Euclidean Distance
  - Isprime
- Array
  - Array Flip
  - 2D Array
  - Calculate Combination Numbers
- Pointer
  - Glance of Memory
  - Pointer Manipulation
  - Two Integer Sort
- Mixed Components
  - Argument Passing
  - Least Common Multiple
  - Merge Two Ordered Arrays

Practical Exercises
- Merge Sort
- Train Station with applying Stack
- Homework3 Problem4
- Sudoku
    - Today's meme

Basic Syntax Exercises

Function

Euclidean Distance

Implement two functions, the first return the Euclidean distance between the origin and (x, y), the second one return the Euclidean distance between pointers (x_1, y_1) and (x_2, y_2).

```
double dist(double x,double y)
{
    //TODO: return the distance between the origin and (x,y).
}
double dist(double x1,double y1,double x2,double y2)
{
    //TODO: return the distance between (x1,y1) and (x2,y2).
}
```

Isprime

Implement a function, return true if the argument integer is a prime number.

```
bool isPrime(int x)
{
    //TODO: check whether x is a prime.
}
```

Array

Array Flip

Input an number n. Then input n more numbers. Store them in an array, reverse the array, and output the result.

```
//Input:
6
1 2 3 4 5 6
//Output:
6 5 4 3 2 1
```

Also try to finish the task without using an array.

2D Array

Read an n by n matrix and calculate its trace (sum of elements on the main diagonal).

```
//input
4
1 2 3 4
2 3 4 5
3 4 5 6
4 5 6 7
//output
16
```

Calculate Combination Numbers

Input n,m and output all combination numbers (a.k.a binomial coefficient) $C(i, j)$ ($0 \leq i \leq n$, $0 \leq j \leq m$).

Pointer

Glance of Memory

```c
#include<stdio.h>
int a[4];
int main()
{
    int e,f,g,h;
    printf("%p %p %p %p\n%p %p %p %p\n",&a[0],&a[1],&a[2],&a[3],&e,&f,&g,&h);
    return 0;
}
```

Conclude that variable are **sometimes** "continuous" in the memory.

Tips: you may receive warning that `&a[0]` doesn't match the type of `%p`. You can either apply a type cast or temporarily remove the flag `-Wall` (which treats all warnings as errors) when compiling your program.

Pointer Manipulation

Get familiar with changing the value of a variable with its address by altering the former program.

Two Integer Sort

Design a function to swap two variables if the former one is bigger than the latter one.

```c
#include<stdio.h>
void sort(/*TODO*/)
{
    //TODO: swap the value of two variable with their address
}
int main()
{
    int a=3,b=2;
    sort(&a,&b);
    printf("%d %d",a,b);//Expect to be 2 3
}
```

And use `sort` function to complete the bubble sort (Always adjust the order of two adjacent components).

```c
for(int i=0;i<n;i++)
    for(int j=/*TODO*/)
        /*TODO*/
```

Mixed Components

Argument Passing

Explain the output of following code.

```c
#include <stdio.h>
void inc1(int a) { ++a; }
void inc2(int *a) { ++*a; }
void inc3(int a[]) { ++a[0]; }
int a, b[20];
int main()
{
    inc1(a);
    printf("%d\n", a);
    inc1(b[0]);
    printf("%d\n", b[0]);
    inc2(&a);
    printf("%d\n", a);
    //inc2(b);
    //printf("%d\n",b[0]);
    //inc3(&a);
    //printf("%d\n",a);
    inc3(b);
    printf("%d\n", b[0]);
}
```

Note: The 4 commented lines involve "implicit conversion" between pointer and array. You don't need to fully understand them now.

Least Common Multiple

Given n integers, find their least common multiple.

The formula you may need: $gcd(a, b) \times lcm(a, b) = ab$.

```
//input
6
1 2 3 4 5 6
//output
60
```

Merge Two Ordered Arrays

Given two sorted integer arrays a and b, merge a into b as one sorted array(stored in a).

You can assume:

1. The number of elements initialized in `a` and `b` are `anum`, `bnum`.
2. Assume that the array a has enough space to hold `anum+bnum` number.

Finish the below function:

```c
void merge(int a[], int b[], int anum, int bnum){
    //TODO: merge a into b as one sorted array(stored in a)
}
```

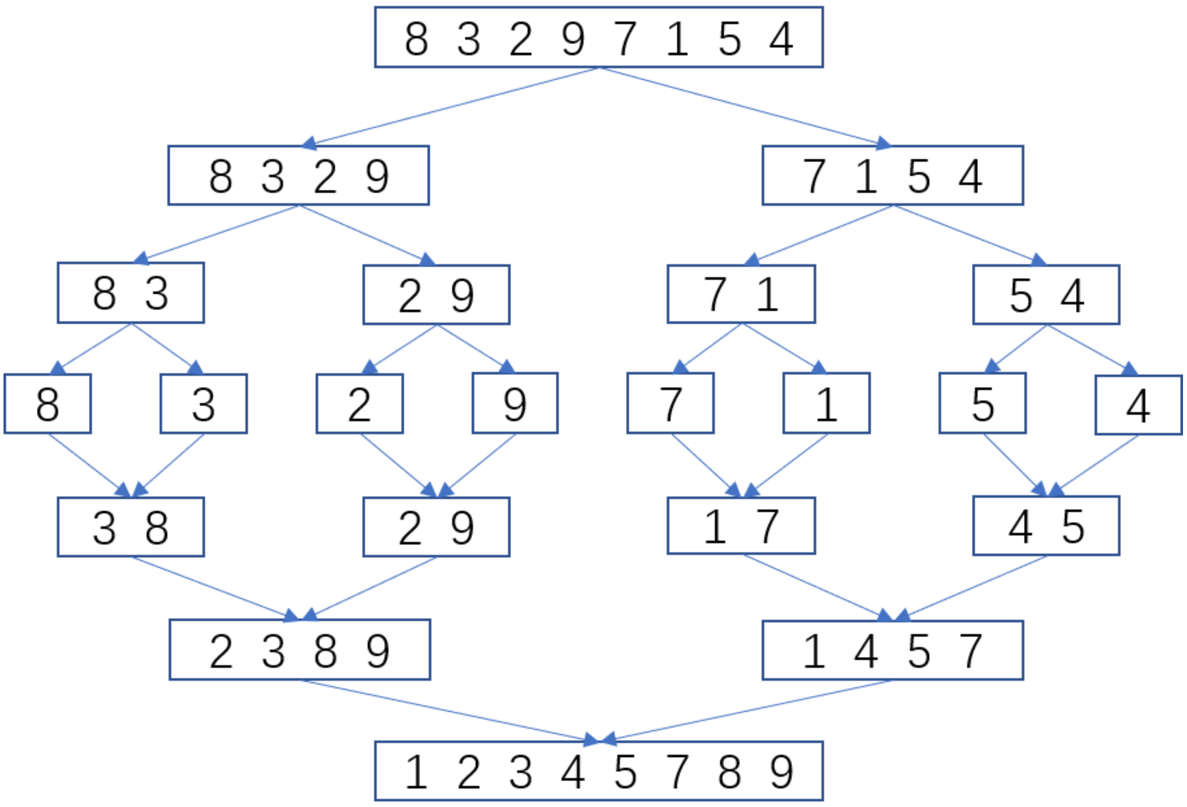
```
Input arguments:
a=[1 3 5 6 0 0 0] anum=4
b=[1 4 7]  bnum=3
Array a after the merge:
a=[1 1 3 4 5 6 7]
```

Practical Exercises

Merge Sort

Recursively use the `merge()` function in the previous part to sort an array.

Hint: a number can be seen as an ordered array with size 1 For example, to sort an array with size 6, from one number to the whole array, a process is something like the procedure below. (Each)



```
void merge_sort(int a[],int length,int b[])
// b is an array for temporary savings.
// sort array a from 0 to length-1
{
    if(length == 1) return ;
    int mid = length / 2;
    //TODO: divide the whole period into two parts and separately handly them.
    merge_sort(a,mid,b);
    merge_sort(&a[mid],length-mid,b[mid]);
    //TODO: let mid to length-1 of b equals mid to length-1 of a.
    for(i=mid;i<length;++i)
        b[i] = a[i];
    //TODO: merge two arrays.
    merge(a,b[mid],mid,length-mid);
}
```

Train Station with applying Stack

A stack is a structure storing data. Stack includes manipulations of adding, showing and popping an element. For a stack, the elements put into the stack later will be popped out earlier.

```c
// define MAXN=1000, EMPTY_VALUE = 0
int num_element;
int stack[MAXN];
// adding
stack[num_element++] = x;
// poping
stack[num_element--] = EMPTY_VALUE;
```

A practical problem Suppose you are in 1900s England and you are a train station officer. You hope to record what trains are stored in one track. The station is a U type station: the train go into the station, stay in the station, and go back in the way to leave the station. Suppose the train can't change tracks, which means if some trains go into the station after the first train, the first train can't leave until the later one moves out. Suggest the station has multiple track. Set a tool (stack) to record the train nearest to the exit for each track.

Input:

The first line includes two numbers: `n`, `m` indicating the total number of tracks and the total number of incidents that you need to handle correspondingly.

Next m lines include the detail of the incidents, each line with two or three numbers `n` `x` and `t`:

- if `n=1`, `x` represents the serial number of the train that moves in; `t` represents the serial number of the track that the train moves in.
- if `n=2`, `x` is absent, and `t` represents the serial number of the track that moves out a train. (you need to deduce the serial number of the train by yourself).

Note:

- the serial number of the track starts from 1, ends at `n`
- the serial number of the train can be any positive integer

Output:

A line including the serial number of the train that is nearest to the exit of each track in the end. If no train exists, output `-1`.

Sample Input:

```
3 6
1 1 1
1 2 1
1 3 3
2 1
1 4 1
1 20 3
```

Sample Output:

```
4 -1 20
```

Homework3 Problem4

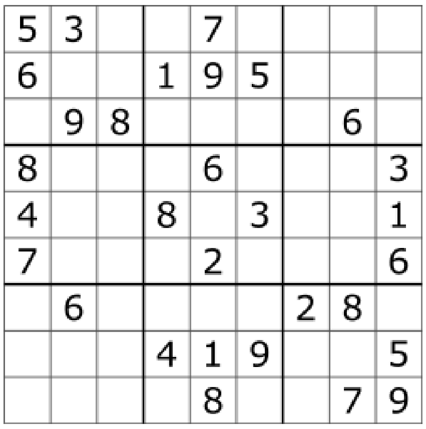
C program usually has a faster speed and steady time compared with MATLAB when there's no many vectors manipulations. So, try to solve your ex3_4 with a C program as a practice of C. You just need to output the minimal sum of time.

```
//input
4
2 12 8 5
4 4 5 12
3 10 8 3
12 0 1 2
//output
10
```

Sudoku

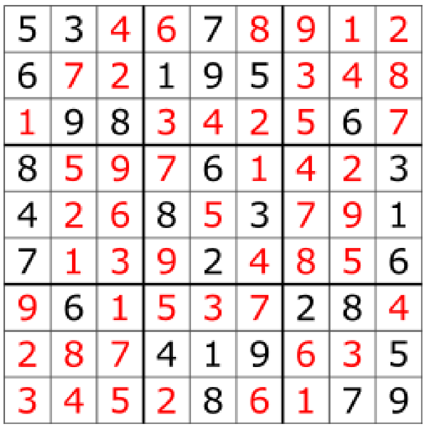
Sudoku is a combinatorial number-placement puzzle. You have to fill a 9×9 grid with digits so that each column, each row, and each of the nine 3×3 subgrids that compose the grid contain all of the digits from 1 to 9.

Now, given a sudoku puzzle, your task is to solve this puzzle.



5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Sample Puzzle



5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Sample Answer

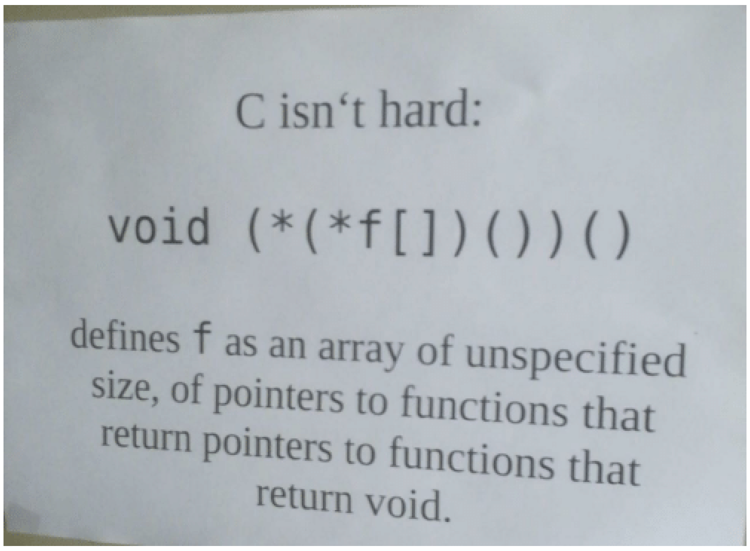
Sample input:

```
0 3 0 0 1 0 0 6 0
7 5 0 0 3 0 0 4 8
0 0 6 9 8 4 3 0 0
0 0 3 0 0 0 8 0 0
9 1 2 0 0 0 6 7 4
0 0 4 0 0 0 5 0 0
0 0 1 6 7 5 2 0 0
6 8 0 0 9 0 0 1 5
0 9 0 0 4 0 0 3 0
```

You should display a matrix representing a completed puzzle.

Today's meme

`#Surprise!`



C isn‘t hard:

```
void (*(*f[])())()
```

defines `f` as an array of unspecified size, of pointers to functions that return pointers to functions that return void.

Found this on the wall of my CS lab, thought Id share