Experiment 8 Function (2)

1. Experimental Purpose

(1) Familiarize yourself further with how to use functions to accomplish specified tasks.

(2) Familiarize yourself with nested and recursive methods of calling functions.

(3) Familiarize yourself with the concepts and uses of global and local variables.

2. Experimental Contents

(1) Write a function that uses the "frothing" method to sort the 10 characters entered in order from small to large.

(a) Input program, compile and run, analysis results.

Code:

```
#include <stdio.h>
int main()
{
    int i,j,t;
    int a[10];
    printf("Enter 10 numbers:\n");
    for(i=0;i<10;i++)
        scanf("%d",&a[i]);
    for(j=0;j<9;j++)
        for(i=0;i<9-j;i++)
            if(a[i]>a[i+1])
            {
                t=a[i];
                a[i]=a[i+1];
                a[i+1]=t;
            }
    printf("\n The 10 numbers sorted are: \n");
    for(i=0;i<10;i++)
        printf("%d    ",a[i]);
    printf("\n");
    return 0;
}
```

(b)Change the string to be sorted to 5, in order from large to small.

Code:

```
#include <stdio.h>
int main()
{
    int i,j,t;
    int a[5];
    printf("Enter 5 numbers:\n");
    for(i=0;i<5;i++)
        scanf("%d",&a[i]);
    for(j=0;j<4;j++)
        for(i=0;i<4-j;i++)
            if(a[i]<a[i+1])
            {
                t=a[i];
                a[i]=a[i+1];
                a[i+1]=t;
            }
    printf("\n The five numbers after sorting are:\n");
    for(i=0;i<5;i++)
        printf("%d    ",a[i]);
    printf("\n");
    return 0;
}
```

（**2**）Converts an integer n to a string recursively. For example, enter 483 and output the string "483". N has indeterminate digits and can be any integer.

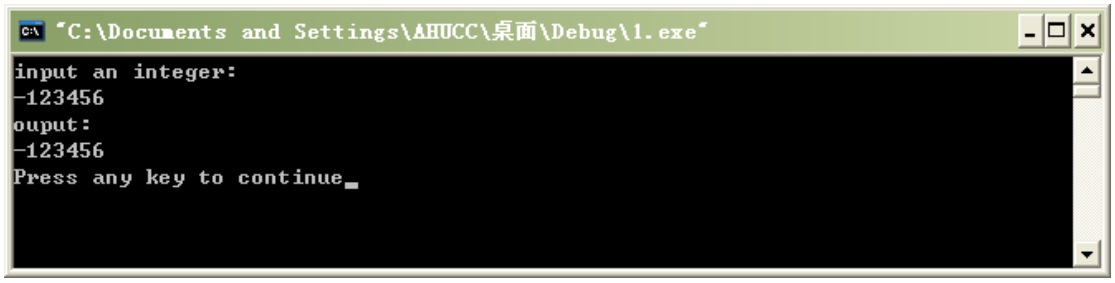
(a) Input program, compile and run, analysis results.

Code:

```
#include <stdio.h>
int main()
{
    void convert(int n);
    int n;
    printf("input an integer:\n");
    scanf("%d",&n);
```

```
    printf("ouput:\n");
    if(n<0)
    {
        putchar('-');
        n=-n;
    }
    convert(n);
    printf("\n");
    return 0;
}

void convert(int n)
{
    int i;
    if((i=n/10)!=0)
        convert(i);
    putchar (n%10+'0');
}
```



(b) Analyse the form and characteristics of recursive calls.

Solution:

Call the function itself directly or indirectly.

A recursive problem can be divided into two phases, backtracking and recursion.

(c)Consider if you can use other methods to solve this problem instead of recursion, and try it on the computer.

```
#include <stdio.h>
int main()
{
    void convert(int n);
    int n;
    printf("input an integer:\n");
    scanf("%d",&n);
    printf("ouput:\n");
    if(n<0)
    {
```

```
            putchar('-');
            n=-n;
        }
        convert(n);
        printf("\n");
        return 0;
}

void convert(int n)
{
        int i;
        char a[10];
        for(i=0;n!=0;)
        {
            ++i;
            a[i]=(n%10+'0');
            n=n/10;
        }
        for(i=i+1;i>=1;)
        {
            --i;
            putchar(a[i]);
        }
}
```

（**3**）Write a function that passes a string from an argument, counts the number of letters, numbers, spaces, and other characters in the string, enters a string in the main function, and outputs the above results.

Use global variables in programs, compile and run programs, and analyze results. Discuss why global variables are used.

Code:

```
#include <stdio.h>
int word=0,digit=0,space=0,other=0;
```

```
int main()
{
    void count(char str[]);
    char str[80];
    printf("Enter a string:\n");
    gets(str);
    printf("string:\n");
    puts(str);
    count(str);
    printf("\nword:%11d\ndogit:%10d\nspace:%10d\nother:%10d\n",word,digit,space,other);
    return 0;
}

void count(char str[])
{   int i;
    for(i=0;str[i]!='\0';i++)
    {
        if((str[i]>='a'&&str[i]<='z')||(str[i]>='A'&&str[i]<='Z'))
            word++;
        else if(str[i]>='0'&&str[i]<='9')
            digit++;
        else if (str[i]==' ')
            space++;
        else
            other++;
    }
}
```

Global variables can be used within each function without having to be declared multiple times to improve programming efficiency.

(b)Whether the program can be modified and run without global variables.

Code:

```
#include <stdio.h>
int main()
{
```

```
    int count(char str[],int x);
    char str[80];
    printf("Enter a string:\n");
    gets(str);
    printf("string:\n");
    puts(str);
    printf("\nword:%11d\ndogit:%10d\nspace:%10d\nother:%10d\n",count(str,1),count(str,2),cou
nt(str,3),count(str,4));
    return 0;
}

int count(char str[],int x)
{   int i;
    int word=0,digit=0,space=0,other=0;
    for(i=0;str[i]!='\0';i++)
    {
        if((str[i]>='a'&&str[i]<='z')||(str[i]>='A'&&str[i]<='Z'))
            word++;
        else if(str[i]>='0'&&str[i]<='9')
            digit++;
        else if (str[i]==' ')
            space++;
        else
            other++;
    }
    if(x==1) return(word);
    else if(x==2) return (digit);
    else if(x==3) return (space);
    else if(x==4) return (other);
}
```

（4）Find the maximum common divisor and the minimum common multiple of two integers, and use a function to find the maximum common divisor. Use another function to find the minimum common multiple based on the maximum common divisor.

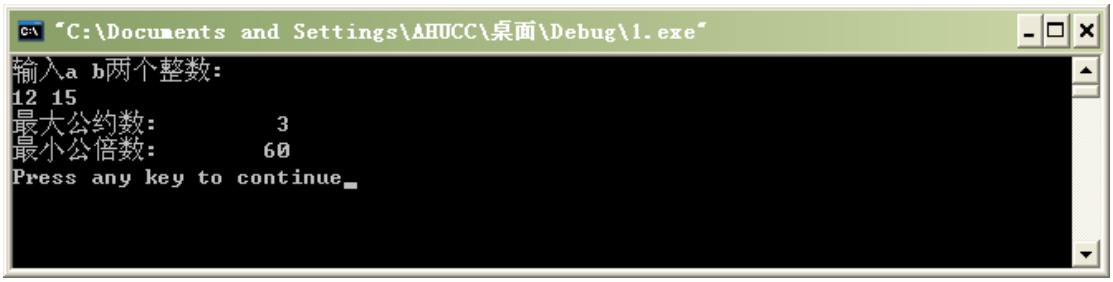
(a) Without global variables, two functions are used to find the maximum common number and the minimum common multiple, respectively. Two integers are input into the main function and passed to the function hcf. The maximum common divisor returns to the main function, and then, together with two integers, is passed to the function LCD as arguments. The minimum common divisor is obtained, and the output of the main function is the maximum common divisor and the minimum common divisor.

Code:

```
#include <stdio.h>
int main()
{
    int hcf(int,int);
    int lcd(int,int);
    int a,b,m,n;
    printf("Enter two integers a b:\n");
    scanf("%d%d",&a,&b);
    m=hcf(a,b);
    n=lcd(m,a*b);
    printf("Maximum common divisor:% 10d\n Minimum common multiple:% 10d\n",m,n);
    return 0;
}

int hcf(int x,int y)
{
    int i;
    for (i=(x<y)?x:y;i>1;i--)
        if((x%i==0)&&(y%i==0))
            break;
        return (i);
}

int lcd(int x,int y)
{
    return(y/x)
}
```



(b)Use the method of global variables. Two global variables are used to represent the maximum common divisor and the minimum common multiple. Use two functions to find the maximum common divisor and the minimum common multiple, but the value is not brought back by the

function, but assigned to the global variable. Output their values in the main function.

Code:

```
#include <stdio.h>
int m,n;
int main()
{
	void hcf(int,int);
	void lcd(int);
	int a,b;
	printf("输入 a b 两个整数:\n");
	scanf("%d%d",&a,&b);
	hcf(a,b);
	lcd(a*b);
	printf("最大公约数:%10d\n 最小公倍数:%10d\n",m,n);
	return 0;
}

void hcf(int x,int y)
{
	for(m=(x<y)?x:y;m>1;m--)
		if((x%m==0)&&(y%m==0))
			break;
}

void lcd(int x)
{
	n=x/m;
}
```