Code and explanation

My code has two parts.

Part 1 shows how the program can read the unlimited-range integers. It is able to extend the size of array and read the input character successfully. It is also able to detect the total amount of integers that has been put in. I tried my best in implementing the unlimited storage of integers. But in the main function, it cannot display the character that has been read in a correct way. I cannot find the reason, although I checked for a long time. Because of this problem, the calculation part should be put separately.

Part 2 shows that the calculation code works very well. The addition, subtraction, and multiplication all works successfully. Here is an example to test this program. The example test is showed below. The subtraction can show a negative result, if a small number minus a big number. In part2, the way of reading integers from user is different from the way in part 1. It has a limitation in reading the input integers. The aim of part 2 is to show that the calculation program works successfully.

Example test:

```
        First integer: 2324
        Second integer: 589
        If the user press ‘+’ , the result is: sum=2913
        If the user press ‘-’ , the result is: minus=1735
        If the user press ‘*’ , the result is: multiplication=1368836
```

Part 1:

```
int GetLongInt(char *& array)
{
        int cnt=0;
        int size1=100;// every time when the array need to be increased, we increase it by 100
        array=(char*)malloc(size1);//allocate memory of array
        int i=0,j=0;
        char c;//c get single char from input, one by one
        c=getchar();
        while(c!='\n')//('0'<=(c=getchar())&&(c=getchar())<='9') ||((c=getchar())=='\\') )
        {
             if (c=='\\')
                {
                        while (getchar()!='\n')
                                ;
                }
                else
                {
                     if (i==size1)
                        {
                                char *temp;
                                temp=(char*)malloc(size1+100);//create a temp
                                for (j=0;j<size1;j++)
                                {
                                        temp[j]=array[j];//copy element from array to temp
                                }
                                free(array);
                                array=temp;//change the position that the pointer points to.
                                size1=size1+100;
                        }
                }
                        array[i] =c;//c is the input of single character. store c into array.
                        i=i+1;
                        cnt++;//total number of elements increases
                        c=getchar();//read the next single character, store it in variable c
        }
        return cnt;
}


int main()
{
        int len1;//the length of var1 is len1
        int i=0;
        char *var1;//create an array to store the first integer
        printf("enter the first integer");
        len1=GetLongInt(var1);// store the input as character to the array var1.
```

```
	for (i=0;i<len1;i++)
	{
		var1[i]=var1[i]-'0';//convert character to integer
	}

	int len2;//the length of var1 is len2
	char *var2;//create an array to store the second integer
	printf("enter the second integer");
	len2=GetLongInt(var2);// store the input as character to the array var2
	for (i=0;i<len2;i++)
	{
		var1[i]=var2[i]-'0';//convert character to integer
	}
```

Part2:

```
#include <stdio.h>
#include <stdlib.h>
int len(char *s);
void sum(int var1[], int var2[], int len1, int len2, int *sum);
int compare(int array1[], int array2[], int len1, int len2) ;
void minus(int var1[], int var2[], int len1, int len2, int *minus);
void BigMinusSmall(int var1[], int var2[], int len1, int len2, int *minus);
void multiplication(int var1[], int var2[], int len1, int len2,int *product);

int len(char *s) //get length of the array of integer
{
	int cnt;
	for(cnt=0; s[cnt]!='\0'; cnt++);
	return cnt;
}

void sum(int var1[], int var2[], int len1, int len2, int *sum)
{
	int MaxIndex1=len1-1;// max index of integer var1
	int MaxIndex2=len2-1;// max index of integer var2
	int i;
	for (i=0; i<len1+len2; i++) //initialization of the "sum" array
	{
		sum[i]=0;
	}
	int c;//c stores the sum of 2 digits
	int Index1=MaxIndex1;
	int Index2=MaxIndex2;
	int IndexSum=len1+len2-1;

	while ((Index1>=0) && (Index2>=0))// as long as both var1 and var2 has digits
	{
		c=var1[Index1]+var2[Index2];//c stores the sum of 2 digits
		sum[IndexSum]=sum[IndexSum]+c%10;//original digit+the remainder
		sum[IndexSum-1]=sum[IndexSum-1]+1.0*c/10;//original digit+how many times the c is for 10
		IndexSum--;
		Index1--;
		Index2--;
	}

	if (Index1!=-1)//when var1 has more digits than var2
	{
		while(Index1>=0)
		{
			c=sum[IndexSum]+var1[Index1];
			sum[IndexSum]=c%10;
			sum[IndexSum-1]=sum[IndexSum-1]+1.0*c/10;
			Index1--;
			IndexSum--;
		}
	}
	else //when var2 has more digits than var1
	{
		while(Index2>=0)
		{
			c=sum[IndexSum]+var2[Index2];
			sum[IndexSum]=c%10;//
			sum[IndexSum-1]=sum[IndexSum-1]+1.0*c/10;
			Index2--;
			IndexSum--;
		}
	}
	printf("sum=");
	IndexSum=0;
	int start;
	while (sum[IndexSum]==0)//the zeros in the beginning of the array should not be presented.
	{
		IndexSum++;// let the index pass over the zeros in the beginning
	}
	start=IndexSum;
	for (IndexSum=start; IndexSum<len1+len2; IndexSum++)
	{
		printf("%d",sum[IndexSum]);//print out the final result of sum
	}
	free(sum);
}

int compare(int array1[], int array2[], int len1, int len2) //compare which integer is larger, in order to know who minus who is reasonable. this means to know array1 => or  < arrray 2
{
	int r=1;
	int i;
	if (len1>len2)
	{
		r=1; //r=1 means len1> len2 means array1>array2
	}
	else if (len1==len2)
	{
		i=0;
		while((array1[i]==array2[i])&&(i<=len1-1))
		{
			i=i+1;
		}

		if (array1[i]>array2[i])
		{
			r=1;//means array1>array2
		}
		else
		{
			r=2;//means array1=<array2
		}
	}
	else
	{
		r=2;//means array1=array2
	}
	return r;
}

void minus(int var1[], int var2[], int len1, int len2, int *minus)//var1-var2
{
	int r;
	r=compare(var1, var2, len1, len2);
	if (r==1)//means var1> or =var2
	{
		BigMinusSmall(var1, var2, len1, len2,minus);
	}
	if (r==2)//means var1<var2
	{
		printf("-");//because small number minus big number
		BigMinusSmall(var2, var1, len2, len1,minus);
	}
}

void BigMinusSmall(int var1[], int var2[], int len1, int len2, int *minus)
{
	int MaxIndex1=len1-1;// get max index of array1
	int MaxIndex2=len2-1;
	if(minus == NULL)
	{
		printf("Error! memory not allocated.");
		exit(0);
	}
	int i;
	for (i=0; i<len1+len2; i++) //initialization of the "sum" array
	{
		minus[i]=0;
	}
	int Index1=MaxIndex1;
	int Index2=MaxIndex2;
	int IndexMinus=len1+len2-1;//get the length of the result of minus

	while ((Index1>=0) && (Index2>=0))//pass the digits in both of the arrays
	{
		if(var1[Index1]>=var2[Index2])//if a big digit minus a small digit
		{
			minus[IndexMinus]=var1[Index1]-var2[Index2];//a digit minus another digit
		}
		else//if a small digit minus a big digit
		{
			minus[IndexMinus]=10+var1[Index1]-var2[Index2];//small digit borrow a 10 from its previous digit
			var1[Index1-1]=var1[Index1-1]-1;// the previous digit gave 1 already
		}
		Index1--;
		Index2--;
		IndexMinus--;
	}
	while(Index1!=-1)
	{
		minus[IndexMinus]=var1[Index1];
		Index1--;
		IndexMinus--;
	}
	IndexMinus=0;
	int start;
	while (minus[IndexMinus]==0)//the zeros in the beginning of the array should not be presented.
	{
		IndexMinus++;
	}
	start=IndexMinus;
	for (IndexMinus=start; IndexMinus<len1+len2; IndexMinus++)
	{
		printf("%d",minus[IndexMinus]);//print the result of minus
	}
	printf("\n\n");
}
```

```
void multiplication(int var1[], int var2[], int len1, int len2,int *product)//multiply var1 with var2
{
        int *temp;//create a temp array. to store the multiplication in each level.
        temp=(int*)malloc((len1+len2)*sizeof(int));//
        int c;
        int MaxIndex1=len1-1;
        int MaxIndex2=len2-1;
        int kmax=len1+len2-1;//k is the index of the product array. k is also the index of temp array
        int    i=0,j=0,k=0,p=0,q;
        for (k=0; k<len1+len2; k++)
        {
                temp[k]=0; //initialize temp as 00000...
                product[k]=0;//product array will add all the temp array together. here, we initialize product as 00000...
        }

        for (j=MaxIndex2; j>=0; j=j-1) //index the var2
        {
                int k=kmax;
                for (p=0; p<MaxIndex2-j; p=p+1) //for (k=len1+len2-1;k>kstart;k=k-1) how many 0 should be put in
                {
                        temp[k]=0;//initialize temp array as 00000...
                        k=k-1;
                }

                for (i=MaxIndex1; i>=0; i=i-1) //index of the var1
                {
                        c=var1[i]*var2[j];//multiply two digits
                        if (i!=len1-1)
                        {
                                c=c+temp[k];//add the number to temp
                        }
                        temp[k]=c%10;//store the remainder of this multiplication to temp[k]
                        temp[k-1]=c/10;//store the quotient of this number
                        k=k-1;
                }
                int d=0;
                for (k=kmax; k>=0; k--)//add the whole level(=temp) to the product
                {
                        d=temp[k]+product[k];
                        product[k]=d%10;
                        product[k-1]=product[k-1]+(d/10);
                }
        }
        printf("multiplication=");
        k=0;
        int start;
        while (product[k]==0)//the zeros in the beginning of the array should not be presented.
        {
```

```
		k++;
	}
	start=k;
	for (k=start; k<len1+len2; k++)
	{
		printf("%d",product[k]);
	}
	printf("\n");
}

int main()
{
	char str1[100];//first array, store input as characters
	int len1;
	int i;
	printf("enter number:");
	scanf("%s",str1);
	while(getchar()!='\n')
		;
	len1=len(str1);
	int var1[len1];//create a new array that store all the elements as integers
	for (i=0; i<len(str1); i++)
	{
		var1[i]=str1[i]-'0';//convert character to integer
	}

	char str2[100];// second array, store input as characters
	int len2;
	printf("enter number:");
	scanf("%s",str2);
	while(getchar()!='\n')
		;
	len2=len(str2);
	int var2[len2];//create a new array that store all the elements as integers
	for (i=0; i<len(str2); i++)
	{
		var2[i]=str2[i]-'0';//convert character to integer
	}

	char operator;
	printf("Enter an operator (+, -, *): ");// ask the user to enter an operator
	scanf("%c", &operator);
	switch (operator)// 4 situations in total. the program do the calculation based on user's command
	{
	case '+':
		printf("addition\n");
		int *SUM;//create an array to store the final result
		SUM=(int*)malloc((len1+len2)*sizeof(int));// allocate the memory
		sum(var1, var2, len1, len2,SUM);//call the function
```

```
            break;

        case '-':
            printf("substraction\n");
            int *MINUS;//create an array to store the final result
            MINUS=(int*)malloc((len1+len2)*sizeof(int));// allocate the memory
            printf("Minus=");
            compare(var1, var2, len1, len2);//call the function
            minus(var1, var2, len1, len2,MINUS);
            break;

        case '*':
            printf("multiplication\n");
            int *PRODUCT;//create an array to store the final result
            PRODUCT=(int*)malloc((len1+len2)*sizeof(int));// allocate the memory
            multiplication(var1, var2, len1,len2, PRODUCT);//call the function
            break;

        default:// not in the previous 3 operate
            printf("Error. This operator does not exist. Please restart the program.\n\n");
            break;
        }

        return 0;
}
```