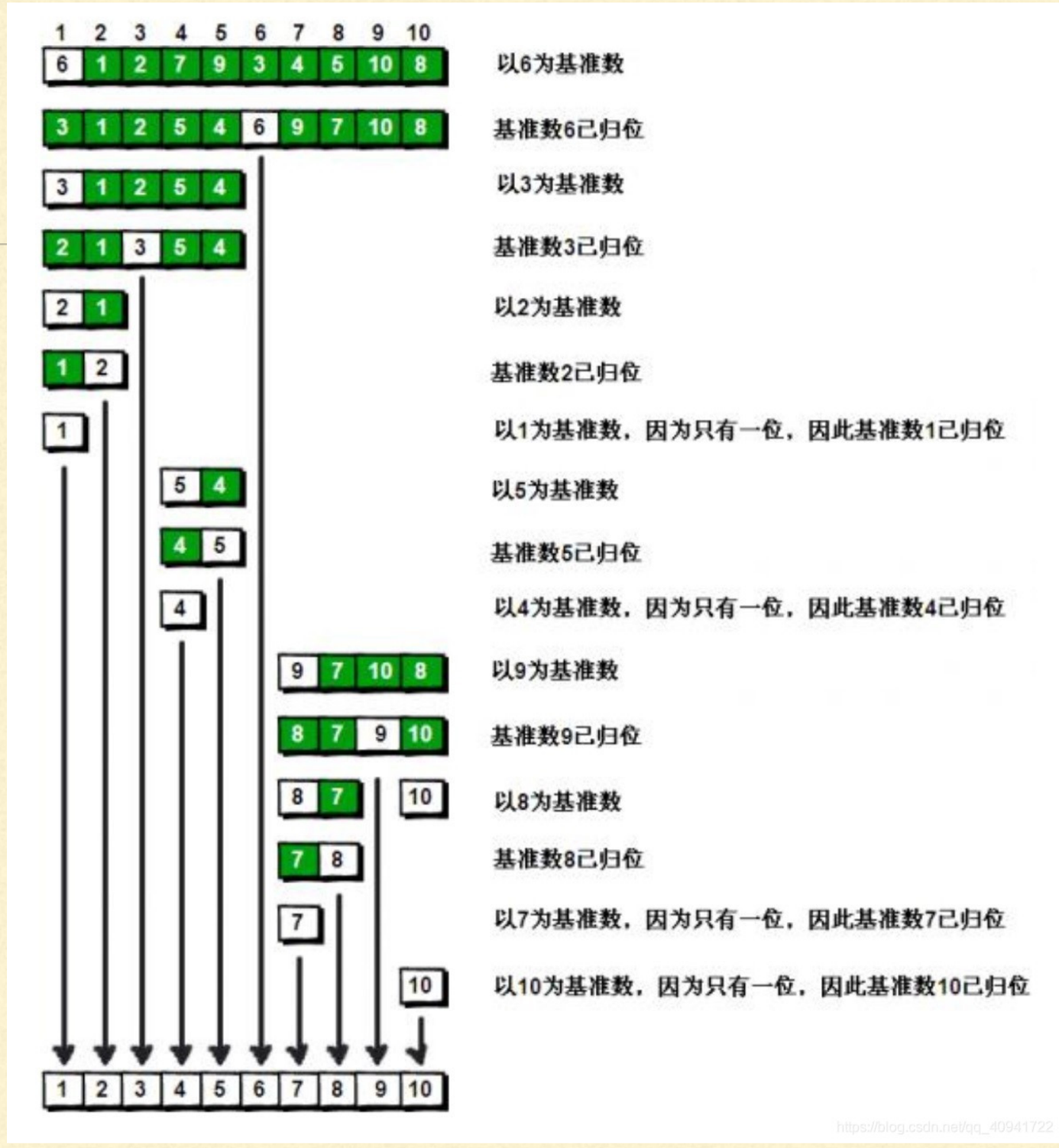
LAB7讲解

2022.11.7

快速排序

https://blog.csdn.net/qq_40941722/article/details/94396010?ops_request_misc=%257B%2522request%255Fid%2522%253A%252216677918741680018651990 9%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request_id=166779187416800186519909&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~top_positive~default-1-94396010-null-null.142^v63^control,201^v3^control_2,213^v1^t3_esquery_v2&utm_term=快速排序&spm=1018.2226.3001.4187



冒泡排序，选择排序，插入排序，快速排序，希尔排序

https://blog.csdn.net/m0_37741420/article/details/106981276?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522166779272016782425191269%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request_id=166779272016782425191269&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~top_positive~default-1-106981276-null-null.142^v63^control,201^v3^control_2,213^v1^t3_esquery_v2&utm_term=选择排序&spm=1018.2226.3001.4187

归并排序

https://blog.csdn.net/DUXS11/article/details/125818272?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522166779281316782412591099%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request_id=166779281316782412591099&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~top_positive~default-1-125818272-null-null.142^v63^control,201^v3^control_2,213^v1^t3_esquery_v2&utm_term=归并排序&spm=1018.2226.3001.4187

归并排序(Merge Sort)是建立在归并操作上的一种既有效又稳定的排序算法，该算法是采用分治法(Divide and Conquer)的一个非常典型的应用。将已有序的子序列合并，得到完全有序的序列。即先使每个子序列有序，再使子序列段间有序。若将两个有序表合并成一个有序表，称为二路归并。

A HAUNTED STORY

```cpp
#include <iostream>
using namespace std;
int main()
{
    int n; //输入的字符串数目
    string s; //输入的字符串
    cin>>n;
    for(int i = 0; i<n;i++)
    {
        cin>>s;
        int l = s.length();  //输入的字符串长度
        cout<<"String #"<<i+1<<endl;
        for(int  j = 0; j<l ;j++)
        {
            if(s[j]=='Z')   //输入的字符串中'Z'用A替换
                cout<<'A';
            else
                cout<<(char)(s[j] + 1);    //输入字符串中的每个字母都按照字母表顺序用下一个字母替换
        }
        cout<<endl<<endl;
    }
    return 0;
}
```

⚠ Implicit conversion loses integer precision: 'std::basic_string<char>::size_type'...

A FUN GAME

```
#include<iostream>
#include <algorithm>
using namespace std;
int main()
{
    char words[101][10],str[10],str1[10];
    int i,j,length1,length2,s=0;
    while(1)    //输入字典
    {
        scanf("%s", words[s]);
        if(strcmp(words[s++],"XXXXXX")==0) break;
    }
    for(i=0; i<s-2; i++)   //按字典序对字典选择排序
        for(j = i+1; j<s-1;j++)
        {
            if(strcmp(words[i],words[j])>0)
            {
                strcpy(str,words[i]);
                strcpy(words[i],words[j]);
                strcpy(words[j],str);
            }
        }
        int flag = 1;
    while(scanf("%s",str)!=EOF && strcmp(str,"XXXXXX")!=0)    //输入待处理的单词
    {
        length1 = strlen(str);                    Implicit conversion loses integer precisi
        sort(str, str+ length1);  //待处理的单词升序排序
        for(i = 0 ;i<s-1;i++)
        {
            length2 = strlen(words[i]);           Implicit conversion loses integer precisi
            strcpy(str1, words[i]);
            sort(str1,str1+length2);  //字典单词按字符升序排序
            if(strcmp(str1 , str)==0)  //输出在字典里存在的单词，设置标志
            {
                printf("%s\n",words[i]);
                flag = 0;
            }

        }
        if(flag)
            printf("NOT A VALID WORD\n");    //字典里不存在相应的单词
        printf("******\n");
    }
    return 0;
}
```

SOMETHING TECHNICAL

```cpp
#include<iostream>
using namespace std;
int n;
char pas[20][200];
char wd1[20],wd2[20];
int main(){
    cin>>n; cin.get();
    for(int i=0;i<n;i++)
        cin.getline(pas[i],200);
    cin>>wd1>>wd2;
    int cnt=0,wdcnt;
    cout<<wd1<<" ";
    for(int i=0;i<n;i++){
        wdcnt=1;
        for(int j=0;j<strlen(pas[i]);j++){
            //cout<<pas[i][j]<<" "<<" "<<cnt<<" "<<wd1[cnt]<<" "<<wdcnt<<endl;
            if(pas[i][j]==' ')
                wdcnt++,cnt=0;
            if(pas[i][j]==wd1[cnt])
            {
                if(cnt==strlen(wd1)-1&&(pas[i][j+1]==' '||j==strlen(pas[i])-1)) cout<<(i+1)<<"-"<<wdcnt<<" ";
                else cnt++;
            }
            else
                cnt=0;

        }

    }
```

Possible misuse of comma operator here

```cpp
    cout<<endl;
    cout<<wd2<<" ";
    cnt=0;
    for(int i=0;i<n;i++)
    {
        wdcnt=1;
        for(int j=0;j<strlen(pas[i]);j++)
        { if(pas[i][j]==' ')
            wdcnt++,cnt=0;
            if(pas[i][j]==wd2[cnt])
            { if(cnt==strlen(wd2)-1&&(pas[i][j+1]==' '||j==strlen(pas[i])-1))
                cout<<(i+1)<<"-"<<wdcnt<<" "; else cnt++;

            }
            else cnt=0;

        }

    } cout<<endl;
    return 0;

}
```

“感谢垂听。”