**实验报告**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 姓名 | | 陈启明 | | 学号 | 2018301040082 | 年级 | 2018级 |
| 成绩  （考核档次或分数） | |  | | 指导教师 | 程媛 | 专业 | 社会科学试验班（信息管理类） |
| 实验类型：  ☑独立实验课  □含实验的理论课  □计划外自选开放实验  □学生自主式开放实验  □大学生科技竞赛 | | | | 实验日期：  20 18 年 上 学期，  第 16 周 | | 实验学时数：  学时 | |
| 相关课程：C programming language | | | | | | | |
| 相关科研项目： | | | | | | | |
| 实验名称 | | | Final project | | | | |
| 1. **预习部分（实验目的、实验基本原理等）**   实验原理：  The principle of the C language  主要仪器设备（含必要的元器件、工具）：  Computer, Dev-C++  实验目的：  To develop the ability of solving practical problem by using the C language, especially the applications with strings and texts.  实验内容：  To design a simple search engine, implement the basic core algorithm and extend it with advanced features like exact search, top search and top k search. The search engine is mainly composed of two search modes, manual guided search and script guided search. All the advanced features are supposed to be applicable for two search modes and compatible with other advanced features.  实验要求：  The tasks include: Implementation of the core search engine, enhancing the core search engine with advanced search features and testing.  Besides, the choice of search mode should be specified by command line parameter, and the output format should meet the requirement. | | | | | | | |
| **二、实验操作部分（可续页）**  1. 实验数据、表格及数据处理  **Source code:**  #include<stdio.h>  #include<string.h>  #include<stdbool.h>  #include<stdlib.h>  #include<direct.h>  void rank();  int manual\_core();  int manual\_exact();  int manual\_top();  int manual\_topk();  int script\_core();  int script\_exact();  int script\_top();  int script\_topk();  void rank(int \*a, int \*b)  {  int x,y,t,t1;  for(y=1;y<=510;y++){  for(x=1;x<=510-y;x++){  if(a[x]<a[x+1]){  t = a[x];  a[x] = a[x+1];  a[x+1] = t;    t1 = b[x];  b[x] = b[x+1];  b[x+1] = t1;  }  }  }  }  int manual\_core()  {  FILE \*p;  bool go\_on = true;  char name[10]; //存放文件名的数组  char str[4096];  char query\_temporary[100];  char query[10][20]; //存放用户检索关键词  char delimiters[] = " \".,;:!?()";  int line = 0;  int i = 0; //query term 的数量  int j = 0;  int id = 0;    printf("Enter the key words, separated by blanks:");  /\*读取用户输入\*/  gets(query\_temporary);    /\*得到query中term的个数\*/  char \*pword\_2 = strtok(query\_temporary,delimiters);  do{  strcpy(query[i],pword\_2);  pword\_2 = strtok(NULL,delimiters);  i++;  }while(pword\_2);    for(id=1;id<=510;id++){  go\_on = true;  sprintf(name,"%03d.txt",id);  p = fopen(name,"r");  while(!feof(p) && go\_on){  fgets(str,1024,p);  if(line >= 5){ //跳过content前的部分  /\*检索\*/  char \*pword\_1 = strtok(str,delimiters);  do{  for(j=0;j<i;j++){  if(!strcmp(pword\_1,query[j])){  printf("%d ",id);  go\_on = false; //修复重复出现id的bug，如果是top搜索就不要用到布尔变量  break;  }  }  pword\_1 = strtok(NULL,delimiters);  }while(pword\_1 && go\_on);  }  line++;  }  fclose(p);  line = 0;  }  return 0;  }  int manual\_exact()  {  FILE \*p;  bool go\_on = true;  char name[10]; //存放文件名的数组  char str[4096];  char query\_temporary[100];  char query[10][20]; //存放用户检索关键词  char delimiters[] = " \".,;:!?()";  int line = 0;  int i = 0; //query term 的数量  int j = 0;  int id = 0;  int c[10] = {0};    printf("Enter the key words, separated by blanks:");  /\*读取用户输入\*/  gets(query\_temporary);    /\*得到query中term的个数\*/  char \*pword\_2 = strtok(query\_temporary,delimiters);  do{  strcpy(query[i],pword\_2);  pword\_2 = strtok(NULL,delimiters);  i++;  }while(pword\_2);        for(id=1;id<=510;id++){  go\_on = true;  sprintf(name,"%03d.txt",id);  p = fopen(name,"r");  while(!feof(p)){  fgets(str,1024,p);  if(line >= 5){ //跳过content前的部分  /\*检索\*/  char \*pword\_1 = strtok(str,delimiters);  do{  for(j=0;j<i;j++){  if(!strcmp(pword\_1,query[j])){  c[j] = 1;  break;  }  }  pword\_1 = strtok(NULL,delimiters);  }while(pword\_1);  }  line++;  }  for(j=0;j<i;j++){  if(c[j] == 0)  break;  }  if(j == i)  printf("%d ",id);  fclose(p);  line = 0;    /\*初始化c\*/  for(j=0;j<i;j++)  c[j] = 0;  }  return 0;  }  int manual\_top()  {  FILE \*p;  bool go\_on = true;  char name[10]; //存放文件名的数组  char str[4096];  char query\_temporary[100];  char query[10][20]; //存放用户检索关键词  char delimiters[] = " \".,;:!?()";  int line = 0;  int i = 0; //query term 的数量  int j = 0;  int k = 0;  int id = 0;  int count = 0;  int max = 0;  int a[511];    printf("Enter the key words, separated by blanks:");  /\*读取用户输入\*/  gets(query\_temporary);    /\*得到query中term的个数\*/  char \*pword\_2 = strtok(query\_temporary,delimiters);  do{  strcpy(query[i],pword\_2);  pword\_2 = strtok(NULL,delimiters);  i++;  }while(pword\_2);        for(id=1;id<=510;id++){  go\_on = true;  sprintf(name,"%03d.txt",id);  p = fopen(name,"r");  while(!feof(p)){  fgets(str,sizeof(str),p);  if(line >= 5){ //跳过content前的部分  /\*检索\*/  char \*pword\_1 = strtok(str,delimiters);  do{  for(j=0;j<i;j++){  if(!strcmp(pword\_1,query[j])){  count++;  break;  }  }  pword\_1 = strtok(NULL,delimiters);  }while(pword\_1);  }  line++;  }  fclose(p);  line = 0;  a[id] = count;  count = 0;  }    /\*找出最大值\*/  for(id=1;id<=510;id++){  if(a[id] > max){  max = a[id];  k = id;  }  }  printf("%d",k);      return 0;  }  int manual\_topk()  {  FILE \*p;  bool go\_on = true;  char name[10]; //存放文件名的数组  char str[4096];  char query\_temporary[100];  char query[10][20]; //存放用户检索关键词  char delimiters[] = " \".,;:!?()";  int line = 0;  int i = 0; //query term 的数量  int j = 0;  int k = 0;  int id = 0;  int count = 0;  int max = 0;  int a[511];  int b[511];    for(i=1;i<=510;i++)  b[i] = i;    printf("Enter the key words, separated by blanks:");  /\*读取用户输入\*/  gets(query\_temporary);    /\*得到query中term的个数\*/  i = 0;  char \*pword\_2 = strtok(query\_temporary,delimiters);  do{  strcpy(query[i],pword\_2);  pword\_2 = strtok(NULL,delimiters);  i++;  }while(pword\_2);    for(id=1;id<=510;id++){  go\_on = true;  sprintf(name,"%03d.txt",id);  p = fopen(name,"r");  while(!feof(p)){  fgets(str,sizeof(str),p);  if(line >= 5){ //跳过content前的部分  /\*检索\*/  char \*pword\_1 = strtok(str,delimiters);  do{  for(j=0;j<i;j++){  if(!strcmp(pword\_1,query[j])){  count++;  break;  }  }  pword\_1 = strtok(NULL,delimiters);  }while(pword\_1);  }  line++;  }  fclose(p);  line = 0;  a[id] = count;  // printf("%d : %d\n",id,a[id]);  count = 0;  }    /\*排序\*/  rank(a,b);    printf("%d ",b[1]);  printf("%d ",b[2]);  printf("%d ",b[3]);      return 0;  }  int script\_core()  {  FILE \*p;  FILE \*p\_1;  FILE \*p\_2;  bool go\_on = true;  char query\_temporary[100];  char query[10][20];  char name[10];  char delimiters[] = " \".,;:!?()";  char str\_1[1024]; //存储脚本  char str[4096]; //存储文本  int line = 0;  int i = 0,j = 0,qid = 1;  int id = 0;  p\_1 = fopen("query.txt","r");    i = 0;  /\*进入大循环体\*/  while(!feof(p\_1)){  fgets(str\_1,sizeof(str\_1),p\_1);  if(str\_1[strlen(str\_1)-1] == '\n')  str\_1[strlen(str\_1)-1] = ' ';  char \*pword\_2 = strtok(str\_1,delimiters);  do{  if(i != 0)  strcpy(query[i-1],pword\_2);  pword\_2 = strtok(NULL,delimiters);  i++;  }while(pword\_2);  /\*检索循环体\*/  for(id=1;id<=510;id++){  go\_on = true;  sprintf(name,"%03d.txt",id);  p = fopen(name,"r");  while(!feof(p) && go\_on){  fgets(str,sizeof(str),p);  if(line >= 5){ //跳过content前的部分  /\*检索\*/  char \*pword\_1 = strtok(str,delimiters);  do{  for(j=0;j<i-1;j++){  if(!strcmp(pword\_1,query[j])){  printf("q%d %03d\n",qid,id);  go\_on = false;  break;  }  }  pword\_1 = strtok(NULL,delimiters);  }while(pword\_1 && go\_on);  }  line++;  }  fclose(p);  line = 0;  }  qid++;  i = 0;  }  fclose(p\_1);  return 0;  }  int script\_exact()  {  FILE \*p;  FILE \*p\_1;  FILE \*p\_2;  bool go\_on = true;  char query\_temporary[100];  char query[10][20];  char name[10];  char delimiters[] = " \".,;:!?()";  char str\_1[1024]; //存储脚本  char str[4096]; //存储文本  int line = 0;  int i = 0,j = 0,qid = 1;  int id = 0;  int c[10] = {0};    p\_1 = fopen("query.txt","r");    /\*进入大循环体\*/  while(!feof(p\_1)){  fgets(str\_1,sizeof(str\_1),p\_1);  if(str\_1[strlen(str\_1)-1] == '\n')  str\_1[strlen(str\_1)-1] = ' ';  char \*pword\_2 = strtok(str\_1,delimiters);  do{  if(i != 0)  strcpy(query[i-1],pword\_2);  pword\_2 = strtok(NULL,delimiters);  i++;  }while(pword\_2);  /\*检索循环体\*/  for(id=1;id<=510;id++){  go\_on = true;  sprintf(name,"%03d.txt",id);  p = fopen(name,"r");  while(!feof(p)){  fgets(str,sizeof(str),p);    if(line >= 5){ //跳过content前的部分  /\*检索\*/  char \*pword\_1 = strtok(str,delimiters);  do{  for(j=0;j<i-1;j++){  if(!strcmp(pword\_1,query[j])){  c[j] = 1;  break;  }  }  pword\_1 = strtok(NULL,delimiters);  }while(pword\_1);  }  line++;  }  for(j=0;j<i-1;j++){  if(c[j] == 0)  break;  }  if(j == i-1){  printf("q%d %03d\n",qid,id);  }    fclose(p);  line = 0;    /\*初始化c\*/  for(j=0;j<i-1;j++)  c[j] = 0;  }  qid++;  i = 0;  }    fclose(p\_1);  return 0;  }  int script\_top()  {  FILE \*p;  FILE \*p\_1;  FILE \*p\_2;  bool go\_on = true;  char query\_temporary[100];  char query[10][20];  char name[10];  char delimiters[] = " \".,;:!?()";  char str\_1[1024]; //存储脚本  char str[4096]; //存储文本  int line = 0;  int i = 0, j = 0, k = 0, qid = 1;  int id = 0;  int id\_1 = 0;  int count = 0;  int max = 0;  int a[511];    p\_1 = fopen("query.txt","r");    i = 0;  /\*进入大循环体\*/  while(!feof(p\_1)){  fgets(str\_1,sizeof(str\_1),p\_1);  if(str\_1[strlen(str\_1)-1] == '\n')  str\_1[strlen(str\_1)-1] = ' ';  char \*pword\_2 = strtok(str\_1,delimiters);  do{  if(i != 0)  strcpy(query[i-1],pword\_2);  pword\_2 = strtok(NULL,delimiters);  i++;  }while(pword\_2);  /\*检索循环体\*/  for(id=1;id<=510;id++){  go\_on = true;  sprintf(name,"%03d.txt",id);  p = fopen(name,"r");  while(!feof(p)){  fgets(str,sizeof(str),p);  if(line >= 5){ //跳过content前的部分  /\*检索\*/  char \*pword\_1 = strtok(str,delimiters);  do{  for(j=0;j<i-1;j++){  if(!strcmp(pword\_1,query[j])){  count++;  break;  }  }  pword\_1 = strtok(NULL,delimiters);  }while(pword\_1);  }  line++;  }  fclose(p);  line = 0;  a[id] = count;  count = 0;  }  /\*找出最大值\*/  for(id\_1=1;id\_1<=510;id\_1++){  if(a[id\_1] > max){  max = a[id\_1];  k = id\_1;  }  }  printf("q%d %03d\n",qid,k);  max = 0;  qid++;  i = 0;  }      fclose(p\_1);  return 0;  }  int script\_topk()  {  FILE \*p;  FILE \*p\_1;  FILE \*p\_2;  bool go\_on = true;  char query\_temporary[100];  char query[10][20];  char name[10];  char delimiters[] = " \".,;:!?()";  char str\_1[1024]; //存储脚本  char str[4096]; //存储文本  int line = 0;  int i = 0, j = 0, k = 0, qid = 1;  int id = 0;  int id\_1 = 0;  int count = 0;  int max = 0;  int a[511]; //存放出现次数  int b[511]; //存放1到510文件ID    for(i=1;i<=510;i++)  b[i] = i;    p\_1 = fopen("query.txt","r");    i = 0;  /\*进入大循环体\*/  while(!feof(p\_1)){  fgets(str\_1,sizeof(str\_1),p\_1);  if(str\_1[strlen(str\_1)-1] == '\n')  str\_1[strlen(str\_1)-1] = ' ';  char \*pword\_2 = strtok(str\_1,delimiters);  do{  if(i != 0)  strcpy(query[i-1],pword\_2);  pword\_2 = strtok(NULL,delimiters);  i++;  }while(pword\_2);  /\*检索循环体\*/  for(id=1;id<=510;id++){  go\_on = true;  sprintf(name,"%03d.txt",id);  p = fopen(name,"r");  while(!feof(p)){  fgets(str,sizeof(str),p);  if(line >= 5){ //跳过content前的部分  /\*检索\*/  char \*pword\_1 = strtok(str,delimiters);  do{  for(j=0;j<i-1;j++){  if(!strcmp(pword\_1,query[j])){  count++;  break;  }  }  pword\_1 = strtok(NULL,delimiters);  }while(pword\_1);  }  line++;  }  fclose(p);  line = 0;  a[id] = count;  count = 0;  }  /\*排序\*/  rank(a,b);  printf("q%d %03d\n",qid,b[1]);  printf("q%d %03d\n",qid,b[2]);  printf("q%d %03d\n",qid,b[3]);  /\*初始化数组b\*/  for(id\_1=1;id\_1<=510;id\_1++)  b[id\_1] = id\_1;    qid++;  i = 0;  }    fclose(p\_1);  return 0;  }  int main(int argc, char \*argv[])  {  if(argc == 2){  if(!strcmp(argv[1], "manual"))  manual\_core();  if(!strcmp(argv[1], "script"))  script\_core();  }    else if(argc == 3){  if(!strcmp(argv[1], "manual")){  if(!strcmp(argv[2], "exactSearch"))  manual\_exact();  if(!strcmp(argv[2], "topSearch"))  manual\_top();  if(!strcmp(argv[2], "topKSearch"))  manual\_topk();  }  else if(!strcmp(argv[1], "script")){  if(!strcmp(argv[2], "exactSearch"))  script\_exact();  if(!strcmp(argv[2], "topSearch"))  script\_top();  if(!strcmp(argv[2], "topKSearch"))  script\_topk();  }  }      else if(argc == 4){  if(!strcmp(argv[1], "manual")){  if(!strcmp(argv[2], "exactSearch"))  manual\_exact();  if(!strcmp(argv[2], "topSearch"))  manual\_top();  if(!strcmp(argv[2], "topKSearch"))  manual\_topk();  printf("\n");  if(!strcmp(argv[3], "exactSearch"))  manual\_exact();  if(!strcmp(argv[3], "topSearch"))  manual\_top();  if(!strcmp(argv[3], "topKSearch"))  manual\_topk();  }  else if(!strcmp(argv[1], "script")){  if(!strcmp(argv[2], "exactSearch"))  script\_exact();  if(!strcmp(argv[2], "topSearch"))  script\_top();  if(!strcmp(argv[2], "topKSearch"))  script\_topk();  printf("\n");  if(!strcmp(argv[3], "exactSearch"))  script\_exact();  if(!strcmp(argv[3], "topSearch"))  script\_top();  if(!strcmp(argv[3], "topKSearch"))  script\_topk();  }  }    else if(argc == 5){  if(!strcmp(argv[1], "manual")){  if(!strcmp(argv[2], "exactSearch"))  manual\_exact();  if(!strcmp(argv[2], "topSearch"))  manual\_top();  if(!strcmp(argv[2], "topKSearch"))  manual\_topk();  printf("\n");  if(!strcmp(argv[3], "exactSearch"))  manual\_exact();  if(!strcmp(argv[3], "topSearch"))  manual\_top();  if(!strcmp(argv[3], "topKSearch"))  manual\_topk();  printf("\n");  if(!strcmp(argv[4], "exactSearch"))  manual\_exact();  if(!strcmp(argv[4], "topSearch"))  manual\_top();  if(!strcmp(argv[4], "topKSearch"))  manual\_topk();  else{  printf("error!");  return 1;  }  }    else if(!strcmp(argv[1], "script")){  if(!strcmp(argv[2], "exactSearch"))  script\_exact();  if(!strcmp(argv[2], "topSearch"))  script\_top();  if(!strcmp(argv[2], "topKSearch"))  script\_topk();  printf("\n");  if(!strcmp(argv[3], "exactSearch"))  script\_exact();  if(!strcmp(argv[3], "topSearch"))  script\_top();  if(!strcmp(argv[3], "topKSearch"))  script\_topk();  printf("\n");  if(!strcmp(argv[4], "exactSearch"))  script\_exact();  if(!strcmp(argv[4], "topSearch"))  script\_top();  if(!strcmp(argv[4], "topKSearch"))  script\_topk();  }  }    else{  printf("error!");  return 1;  }    return 0;  }  2. 实验操作过程（可用图表示）  The whole program is composed of a main function and nine subfunctions, which include manual\_core, manual\_exact, manual\_top, manual\_topk, script\_core, script\_exact, script\_top, script\_topk and rank. Among them, the rank function is used for top k mode. In the main function, I use command line parameters and decisions making statements to combine all the features together (as specified in the source code).  **Flow chart**   1. manual core      1. manual exact      1. manual top      1. manual top k      1. script core      1. script exact     (7) script top    (8) script top k    3. 结论  The program generally meets the requirement and works successfully. | | | | | | | |
| **三、实验效果分析（包括仪器设备等使用效果、实验完成情况）**  **1. manual mode**  (1) core    (2) exact    (3) top    (4) top K    **2. script mode**  (1) core    (2) exact    (3) top    (4) top K    **3. displaying the compatible features (some examples)**      **4. some defects**  (1) My source code is over 700 lines so it appears too lengthy, which makes it difficult to read.  (2) There are many identical parts in my program, which imply that the quality of the source code is not high enough.  (3) Fail to take full advantages of functions.  **5. summary**  To conclude, I have a great sense of achievement when I complete the final project, although the process is hard. It took me a long period of time to make it and fix the bugs.  Actually, the pseudo code of the core search engine benefits me a lot, which helps me get a clear thinking and advance the search engine with different modes. After a long thinking, I apply the “counter” as well as bubble sort to my top search and top k search, and develop the algorithm of exact search by using another array to record whether each of query terms is found in the document.  I’ve had a basic understanding of the C language, which lays the foundation for learning other programming language. However, learning knowledge only in class is not enough, I think I have intense interest to know more about programming language. Maybe it seems difficult but I’ll try my best to make it. | | | | | | | |
| **教师评语** |  | | | | | | |