# 程序概况

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include "quechain.h"  int main() {  puts("请选择题目：");  puts(" 1.C语言");  puts(" 2.英语");  puts(" 3.日语");  puts(" 4.高数");  int \_userInput;  scanf("%d", &\_userInput);  getchar();// 过滤回车（ASCII：10）  if (3 == \_userInput) {  puts("您选择了日语，现在开始答题");  puts("夜は何( )食べましたか。");  puts("A が");  puts("B を");  puts("C も");  puts("D は");  }  char cUserAnser[10];  gets(cUserAnser);  puts("您历史最高得分是：100分");  puts("您本次得分是：80分，错：1题，对：4题");  puts("按任意键查看错题");  getchar();  puts("夜は何( )食べましたか。");  puts("A が");  puts("B を");  puts("C も");  puts("D は");  puts("正确答案是：B 您的答案是：A");  return 0;  } |
| 请选择题目：  1.C语言  2.英语  3.日语  4.高数  3  您选择了日语，现在开始答题  夜は何( )食べましたか。  A が  B を  C も  D は  A  您历史最高得分是：100分  您本次得分是：80分，错：1题，对：4题  按任意键查看错题  夜は何( )食べましたか。  A が  B を  C も  D は  正确答案是：B 您的答案是：A  -------------------------------- |

# 文件操作

## 读

int fscanf(FILE \*fp, const char \*format, ...)

从文件读取字符串，遇到 空格/换行/制表符 时停止

头文件<stdio.h>

|  |
| --- |
| void readFile(char \*path) {  FILE \*fp;  char buff[255];  fp = **fopen**(path, "r");  **//int fscanf(FILE \*fp, const char \*format, ...)**  **//从文件读取字符串，遇到 空格/换行/制表符 时停止**  **fscanf(fp, "%s", buff);**  printf("读入 : %s\n", buff );  **fclose**(fp);  } |

char \*fgets( char \*buf, int n, FILE \*fp );

从 fp所指输入流中读取n-1个字符；如果文件中的该行不足n-1个字符，则读完该行就结束

头文件<stdio.h>

|  |
| --- |
| void readFile(char \*path) {  FILE \*fp;  char buff[3];  fp = fopen(path, "r");  **//char \*fgets( char \*buf, int n, FILE \*fp );**  **//从 fp所指输入流中读取n-1个字符；如果文件中的该行不足n-1个字符，则读完该行就结束**  **//把读取的字符串复制到缓冲区 buf，**  **fgets(buff, 3, (FILE\*)fp);**  **printf("读入: %s\n", buff );**  **printf("%d\n", strlen(buff) );**  fclose(fp);  } |
| 全文读完 |
| void readFile(char \*path) {  FILE \*fp;  char buff[300];  fp = fopen(path, "r");  char \*s = NULL;  int nLine = 1;  do {  s = fgets(buff, sizeof(buff), (FILE\*)fp);  printf("%d: %s\n",nLine++, s );  } while(s != NULL);  fclose(fp);  } |

### 读选择题

|  |
| --- |
| 选择题示例：  #题目（一个#）  A 选项A（A~D加空格）  B 选项B  C 选项C  D 选项D  ##标答（两个#） |
| #include <stdio.h>  #include <stdlib.h>  #include <string.h>  void readFile(char \*path) {  FILE \*fp;  char buff[300];  fp = fopen(path, "r");  char \*s = NULL;  int nLine = 1;  do {  s = fgets(buff, sizeof(buff), (FILE\*)fp);  **if(s==NULL || strlen(s) <3) {**  **continue;**  **}**  **char\* sOption=NULL;**  **if ('#'==s[0] && s[1]!='#') {**  **printf("%d: 题目：%s\n",nLine++, s );**  **} else if('#'==s[0] && '#' == s[1]) {**  **printf("%d: 标答：%s\n",nLine++, s );**  **} else if((sOption = strstr(s,"A "))!=NULL) {**  **printf("%d: 选项：%s\n",nLine++, sOption );**  **} else if((sOption = strstr(s,"B "))!=NULL) {**  **printf("%d: 选项：%s\n",nLine++, sOption );**  **} else if((sOption = strstr(s,"C "))!=NULL) {**  **printf("%d: 选项：%s\n",nLine++, sOption );**  **} else if((sOption = strstr(s,"D "))!=NULL) {**  **printf("%d: 选项：%s\n",nLine++, sOption );**  **}**  } while(s != NULL);  fclose(fp);  }  int main(int argc, char \*argv[]) {  char sDataPath[200];  readFile("D:\\ExamData\\西游记.txt");  return 0;  } |

### 使用结构体封装数据

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #define MAX\_LEN 500  **struct Question {**  **char title[MAX\_LEN];**  **char optA[MAX\_LEN];**  **char optB[MAX\_LEN];**  **char optC[MAX\_LEN];**  **char optD[MAX\_LEN];**  **char answer[5];**  **};**  void printQuestion(struct Question q) {  printf("%s\n", q.title);  printf(" %s\n", q.optA);  printf(" %s\n", q.optB);  printf(" %s\n", q.optC);  printf(" %s\n", q.optD);  printf("标答%s\n", q.answer);  }  void readFile(char \*path) {  FILE \*fp;  char buff[MAX\_LEN];  fp = fopen(path, "r");  char \*s = NULL;  int nLine = 1;  do {  s = fgets(buff, sizeof(buff), (FILE\*)fp);  if(s==NULL || strlen(s) <3) {  continue;  }  **struct Question q;**  char\* sOption=NULL;  if ('#'==s[0] && s[1]!='#') {  strcpy(q.title, s+1); //去标记  } else if('#'==s[0] && '#' == s[1]) {  strcpy(q.answer, s+2);  } else if((sOption = strstr(s,"A "))!=NULL) {  strcpy(q.optA, sOption);  } else if((sOption = strstr(s,"B "))!=NULL) {  strcpy(q.optB, sOption);  } else if((sOption = strstr(s,"C "))!=NULL) {  strcpy(q.optC, sOption);  } else if((sOption = strstr(s,"D "))!=NULL) {  strcpy(q.optD, sOption);  }  **printQuestion(q);**  } while(s != NULL);  fclose(fp);  }  int main(int argc, char \*argv[]) {  char sDataPath[200];  readFile("D:\\ExamData\\西游记.txt");  return 0;  } |

# 链表

## 头文件：quechain.h

|  |
| --- |
| struct Question {  int \_id;  struct Question\* pre;  struct Question\* next;  };  void chain\_print(struct Question\* qFirst);  void chain\_add(struct Question\* qFirst, struct Question\* qAdd);  void chain\_remove(struct Question\* qFirst, struct Question\* qRemove);  struct Question\* chain\_get(struct Question\* qFirst, int index);  int chain\_count(struct Question\* qFirst);  struct Question\* newQuestion(int id);  void newQuestion2(int id, struct Question\*\* q); |

## 源文件：quechain.c

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include "quechain.h"  void chain\_print(struct Question\* qFirst) {  puts("----------print------------");  struct Question\* q = qFirst;  if (qFirst->next == NULL) {  puts("没有元素可以打印");  return;  } else {  q = qFirst->next;  }  // 遍历链表  for(q; (q->next) != NULL; q=q->next ) {  printf("%d\n", q->\_id);  }  // 最后一项特殊对待  printf("%d\n", q->\_id);  }  void chain\_add(struct Question\* qFirst, struct Question\* qAdd) {  // 遍历链表  struct Question\* q = qFirst;  for(q; (q->next) != NULL; q=q->next ) {  }  // 最后一项  q->next = qAdd;  qAdd->pre = q;  }  void chain\_remove(struct Question\* qFirst, struct Question\* qRemove) {  struct Question\* qPre = NULL;  struct Question\* qNext = NULL;  struct Question\* q = qFirst;  if (qFirst == qRemove) {  (qFirst->next)->pre = NULL;  qFirst = (qFirst->next);  free(q);  return;  }  // 遍历链表  for(q; (q->next) != NULL; q=q->next ) {  if (q == qRemove) {  qPre = q->pre;  qNext = q->next;  if (qPre!=NULL) {  qPre->next= qNext;  }  if (qNext!=NULL) {  qNext->pre = qPre;  }  free(qRemove);  return;  }  }  // 最后一项  if (q == qRemove) {  qPre = q->pre;  if (qPre!=NULL) {  qPre->next= qNext;  }  free(qRemove);  }  }  struct Question\* chain\_get(struct Question\* qFirst, int index) {  int i = 0;  // 遍历链表  struct Question\* q = qFirst;  for(q;  (q->next) != NULL;  q=q->next,i++ ) {  if (index == i) {  return q;  }  }  if (index == i) {  // 获取最后一个元素  return q;  }  return NULL;  }  int chain\_count(struct Question\* qFirst) {  int i = 0;  // 遍历链表  struct Question\* q = qFirst;  for(q;  (q->next) != NULL;  q=q->next,i++ ) {  }  return i;  }  struct Question\* newQuestion(int id) {  struct Question\* q = (struct Question\*)malloc(sizeof(struct Question));  memset(q, 0, sizeof(struct Question));  q->\_id = id;  return q;  }  void newQuestion2(int id, struct Question\*\* q) {  \*q = (struct Question\*)malloc(sizeof(struct Question));  memset(\*q, 0, sizeof(struct Question));  (\*q)->\_id = id;  } |

## 测试文件：testc.c

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include "quechain.h"  int main() {  struct Question\* qFirst = NULL;  qFirst = newQuestion(0);  struct Question\* q1 = NULL;  newQuestion2(1,&q1);  chain\_add(qFirst, q1);  struct Question\* q2 = newQuestion(2);  chain\_add(qFirst, q2);  struct Question\* q3 = newQuestion(3);  chain\_add(qFirst, q3);  struct Question\* q4 = newQuestion(4);  chain\_add(qFirst, q4);  struct Question\* q5 = newQuestion(5);  chain\_add(qFirst, q5);  chain\_print(qFirst);  printf("get %d\n", chain\_get(qFirst, 1)->\_id);  printf("get %d\n", chain\_get(qFirst, 3)->\_id);  printf("get %d\n", chain\_get(qFirst, 5)->\_id);  //------------------------------  chain\_remove(qFirst, q3);  chain\_print(qFirst);  chain\_remove(qFirst, q5);  chain\_print(qFirst);  chain\_remove(qFirst, q1);  chain\_print(qFirst);  printf("元素个数: %d\n", chain\_count(qFirst));  return 0;  } |

# 随机抽题

## 随机数

|  |
| --- |
| **srand**(time(0));  int i;  for(i = 0; i<20; i++) {  printf("%d ",**rand**()%10+1);  } |