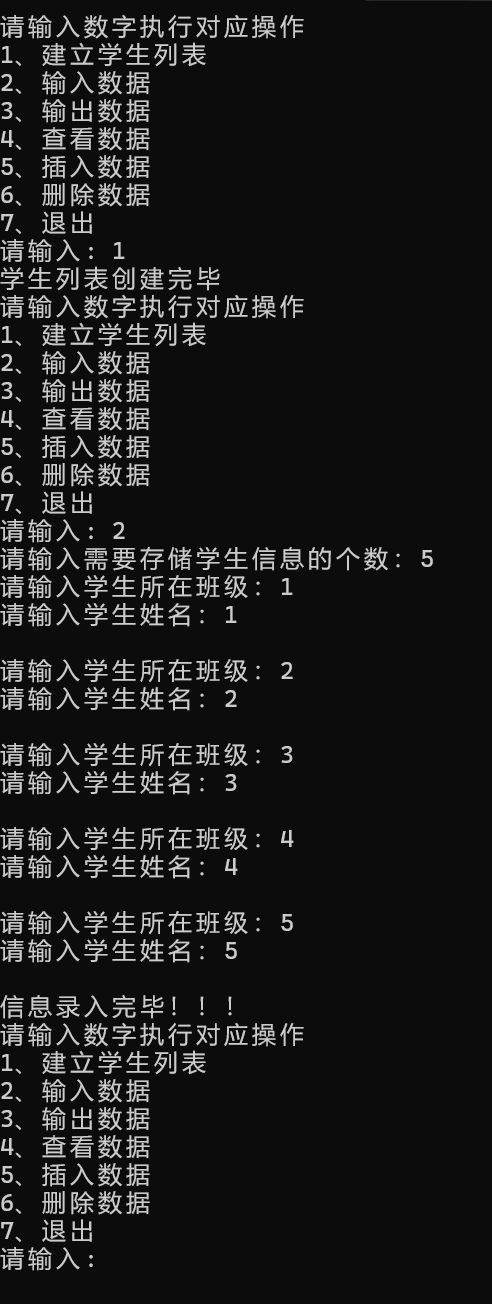
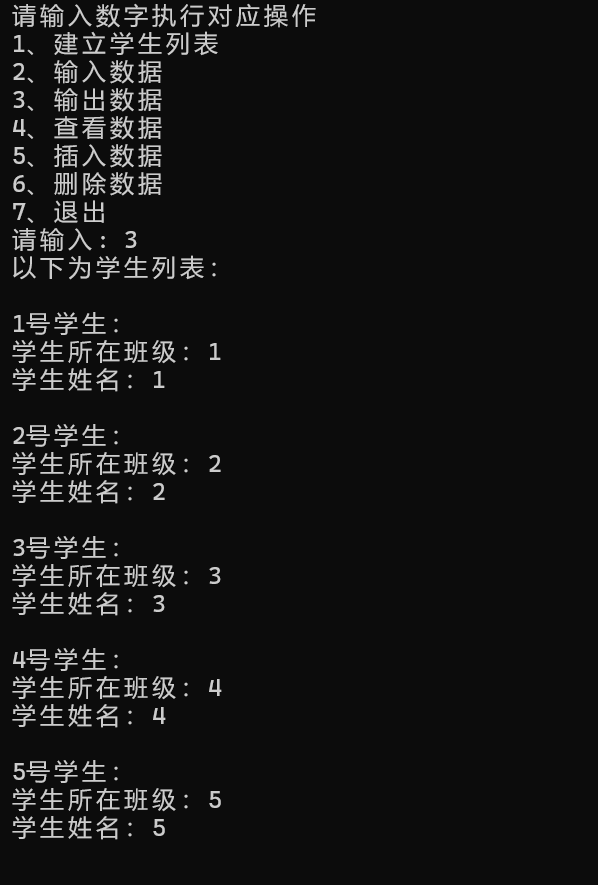
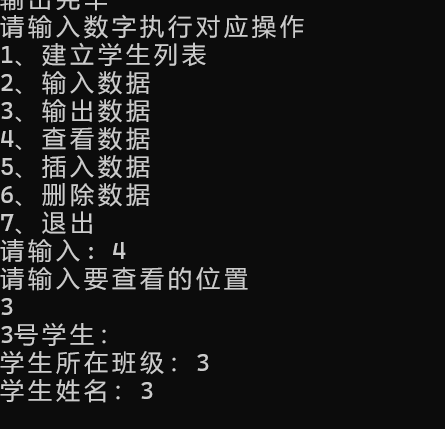
顺序表的建立与输入操作：



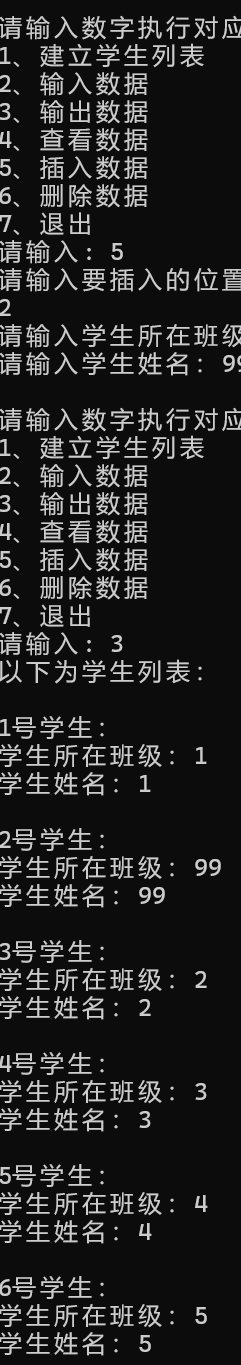
顺序表的输出操作：



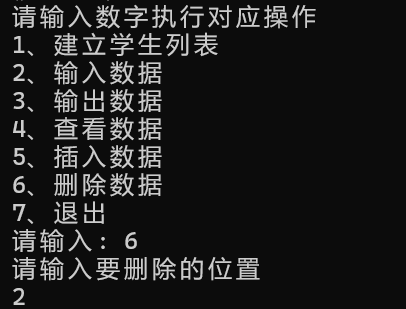
顺序表的查找操作：



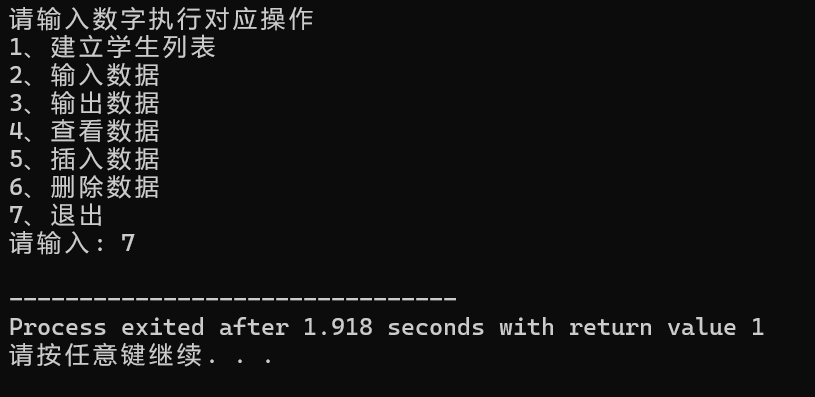
插入并输出数据操作：



删除操作：



退出操作：



代码展示：

#include <iostream>

#include <string>

#define OK 1

#define ERROR 0

#define Maxsize 10

using namespace std;

typedef struct

{

int stu\_class;

string name;

} student;

typedef struct

{

student \*pstu;

int length;

}Student\_list;

inline int Create\_student(Student\_list &p) // n表示创建的顺序表最大长度

{

p.pstu= new student[Maxsize];

p.length = 0;

if(!p.pstu)

{

cout << "学生列表创建失败，空间不足" << endl;

return ERROR;

}

cout << "学生列表创建完毕" << endl;

return OK;

}

int Input\_student(Student\_list &p) //输入数据

{

if(!p.pstu)

{

cout << "学生列表未建立，请重新操作" << endl;

return ERROR;

}

int n = 0;

int i = p.length;

int memory = p.length;

cout << "请输入需要存储学生信息的个数：";

while(1)

{

cin >> n;

if(p.length+n > Maxsize)

{

cout << "超出最大存储空间" << endl;

cout << "请重新输入: ";

continue;

}

for(; i < (memory + n) ; ++i)

{

cout << "请输入学生所在班级：";

cin >> p.pstu[i].stu\_class;

cout << "请输入学生姓名：";

cin >> p.pstu[i].name;

cout << endl;

p.length++;

}

if(i == (memory + n))

{

cout << "信息录入完毕！！！" << endl;

break;

}

}

return OK;

}

int Output\_student(Student\_list &p) //输出数据

{

if(!p.pstu)

{

cout << "学生列表为空或不存在，请先创建列表" << endl;

return ERROR;

}

cout << "以下为学生列表：" << endl << endl;

for(int i = 0 ; i < p.length ; ++i)

{

cout << i+1 << "号学生：" << endl;

cout << "学生所在班级：";

cout << p.pstu[i].stu\_class << endl;

cout << "学生姓名：";

cout << p.pstu[i].name << endl;

cout << endl;

}

cout << endl << "输出完毕" << endl;

return OK;

}

int Insert\_student(Student\_list &p , int location) //插入数据//location表示要插入的位置

{

if(p.length+1 > Maxsize || location > Maxsize || location < 1)

{

cout << "当前空间不足以插入数据或插入位置不合法" << endl;

return ERROR;

}

else

{

for(int i = p.length-1 ; i >= location - 1 ; --i)

{

p.pstu[i+1].name = p.pstu[i].name;

p.pstu[i+1].stu\_class = p.pstu[i].stu\_class;

}

cout << "请输入学生所在班级：";

cin >> p.pstu[location-1].stu\_class;

cout << "请输入学生姓名：";

cin >> p.pstu[location-1].name;

cout << endl;

p.length++;

return OK;

}

}

int Delete\_student(Student\_list &p , int location)

{

if(location < 1 || location > p.length)

{

cout << "删除位置不合法" << endl;

return ERROR;

}

for(int i = location ; i <= p.length-1 ; ++i)

{

p.pstu[i-1].name = p.pstu[i].name;

p.pstu[i-1].stu\_class = p.pstu[i].stu\_class;

}

p.length--;

return OK;

}

inline int Check\_student(Student\_list &p , int location)

{

if(!p.pstu)

{

cout << endl << "学生列表为空或不存在，请重新操作" << endl;

}

cout << location << "号学生：" << endl;

cout << "学生所在班级：";

cout << p.pstu[location-1].stu\_class << endl;

cout << "学生姓名：";

cout << p.pstu[location-1].name << endl;

cout << endl;

}

int main()

{

int num = 0;

int loc = 0;

Student\_list stu;

while(1)

{

int loc = 0;

cout << "请输入数字执行对应操作" << endl;

cout << "1、建立学生列表" << endl;

cout << "2、输入数据" << endl;

cout << "3、输出数据" << endl;

cout << "4、查看数据" << endl;

cout << "5、插入数据" << endl;

cout << "6、删除数据" << endl;

cout << "7、退出" << endl;

cout << "请输入: ";

cin >> num;

switch(num)

{

case 1:

Create\_student(stu);

break;

case 2:

Input\_student(stu);

break;

case 3:

Output\_student(stu);

break;

case 4:

cout << "请输入要查看的位置" << endl;

cin >> loc;

Check\_student(stu , loc);

break;

case 5:

if(!stu.pstu)

{

cout << endl << "学生列表为空或不存在，请重新操作" << endl;

}

cout << "请输入要插入的位置" << endl;

cin >> loc;

Insert\_student(stu , loc);

break;

case 6:

if(!stu.pstu)

{

cout << endl << "学生列表为空或不存在，请重新操作" << endl;

}

cout << "请输入要删除的位置" << endl;

cin >> loc;

Delete\_student(stu , loc);

break;

case 7:

exit(1);

}

}

}