**孔迪 2016010908009**

**上机实验一**

**ex基础训练**

table\_t \* init\_table()

{

if (table\_t \*newt = (table\_t\*)malloc(sizeof(table\_t)))

{

newt->length = 0;

return newt;

}

else

return 0;

}

void free\_table(table\_t \* t)

{

free(t);

}

int get\_table(table\_t \* table,int index, element\_t \* elem)

{

//判断index是否超出顺序表范围

if(index <= 0 || index > table->length){

return -1;

}

//复制元素内容到指定空间中;

memcpy(elem,&(table->data[index]),sizeof(element\_t));//error

return 0;

}

int add\_table(table\_t \* table, element\_t data)

{

if(table->length >= MAX\_TABLE\_SIZE)

return -1;

else

{

memcpy(&table->data[table->length], &data, sizeof(element\_t));

table->length++;

return 0;

}

}

int insert\_table(table\_t \* table, element\_t data, int location)

{

int i;

if (location > MAX\_TABLE\_SIZE || location < 1 || table->length >= MAX\_TABLE\_SIZE)

return -1;

else if (location > table->length)

{

add\_table(table, data);

return 0;

}

else

{

for (i = table->length; i >= location; i--)

memcpy(&table->data[i], &table->data[i - 1], sizeof(element\_t));

memcpy(&table->data[i], &data, sizeof(element\_t));

table->length++;

return 0;

}

}

int insert\_table\_by\_order(table\_t \* table, element\_t data)

{

if (table->length >= MAX\_TABLE\_SIZE)

return -1;

else

{

int i = 0;

while (data.stuID > table->data[i].stuID && i < table->length)

{

i++;

}

insert\_table(table, data, i + 1);

return 0;

}

}

int delete\_table(table\_t \* table, char \* name)

{

for (int i = 0; i < table->length; i++)

{

if (strcmp(table->data[i].stuName, name) == 0)

{

table->length--;

for (int j = i; j < table->length;j++)

memcpy(&table->data[j], &table->data[j + 1], sizeof(element\_t));

return 0;

}

}

return -1;

}

void delete\_table\_below(table\_t \* table, int x)

{

for (int i = 0; i < table->length; i++)

{

if (table->data[i].overall < x)

{

table->length--;

for (int j = i; j < table->length; j++)

memcpy(&table->data[j], &table->data[j + 1], sizeof(element\_t));

i--;

}

}

return ;

}





