Data Structure 01

MKQ

September 5, 2019

Contents

1 抽象数据类型 ADT

Abstract Data Type

- D: 数据对象
- S: 数据之间的关系
- T: 对于数据的操作

1.1 基本操作的定义格式

```
操作名()
{
    初始条件<>
    操作结果<>
}
```

1.2 举例, 复数

```
ADT COMPLEX
{
    数据对象:D={e1,e2|R}
    数据关系:R1={<e1,e2>}
    基本操作:
InitComplex(&z,v1,v2)
//创建一个复数
GetReal(z,&realPart)
//获取一个复数的实部
GetImage(z.&imagPart)
//获取一个复数的虚部
}ADT Complex
```

1.3 抽象数据类型涉及的问题

- 运算集合的定义
- 抽象数据类型在机内的存储
- 运算集合 u 如何被编程实现
- 这玩意是给人看的 (使用者, 实现者)

于是这一章就完结啦, 撒花 ~~~~~

2 类 C 语言介绍

只是一种概念上的语言, 编译不能通过的… 但是这个语法类似 C, 为了突出运算规则的描述

- 提升了可读性
- 考试的时候可以写 C 也可以写类 C
- 有点害怕…万一学不会就凉了
- 一些约定

```
#define TRUE 1
#define FALSE 0
```

```
#define OK 1
#define ERROR 0
#defin
e INFEASTBLE -1 //不可实现

#define OVERFLOW -2
typedef int status;
```

- 同时还约定, 数据元素的类型名是 ElemType
- 任何变量允许不定义直接使用
- 操作算法描述

```
Return_value func_name(args)
{
    //introductions
    statements;
}
```

- 结束语句
  - 函数结束: return;return();
  - case 结束: break;
  - 错误退出: exit(错误代码);
- 基本函数
  - max()
  - min()
  - abs()
  - floor()//向下取整
  - ceil()//向上取整
  - eof()//文件是否结束
  - eoln()//行是否结束
- 内存分配
  - 指针变量 =new 数据类型;
  - malloc

- realloc(旧的基地址, 新分配的长度)

其实就是重新申请了一块空间然后复制过去

- delate
- free()

  int a[5]; 这样的数组是不能扩容的, 因为 a 是一个指针常量不能重新赋值, 它的空间在编译时便已经被分配好了的, 但是 malloc 的空间是在运行时分配的

- cpp 中的引用变量

```
int x=10;
int &rx=x,&rrx=rx;
//引用变量在定义时必须赋值,可以看做变量的别名
//但把它搞到函数里面的话,函数可以改变变量的值
//引用变量被初始化后就不能更改它的值了
//函数参数表定义时可以在引用参数前加&,这样编译器就知道这是个引用变量
void add(int x,int &y)
{
    x=x+1;
    y=y+1;
}
//然后发现y的值变化了,但是x的值没变
```