数据结构和算法

作者：小甲鱼

让编程改变世界

Change the world by program

线索二叉树

- 为什么需要线索二叉树呢？
- 我想正如程序猿发觉单链表并不总能满足他们设计的程序某些要求的时候，发明了双向链表来弥补一样，线索二叉树也是在需求中被创造的！
- 那普通的二叉树到底有什么缺陷让我们发指呢？

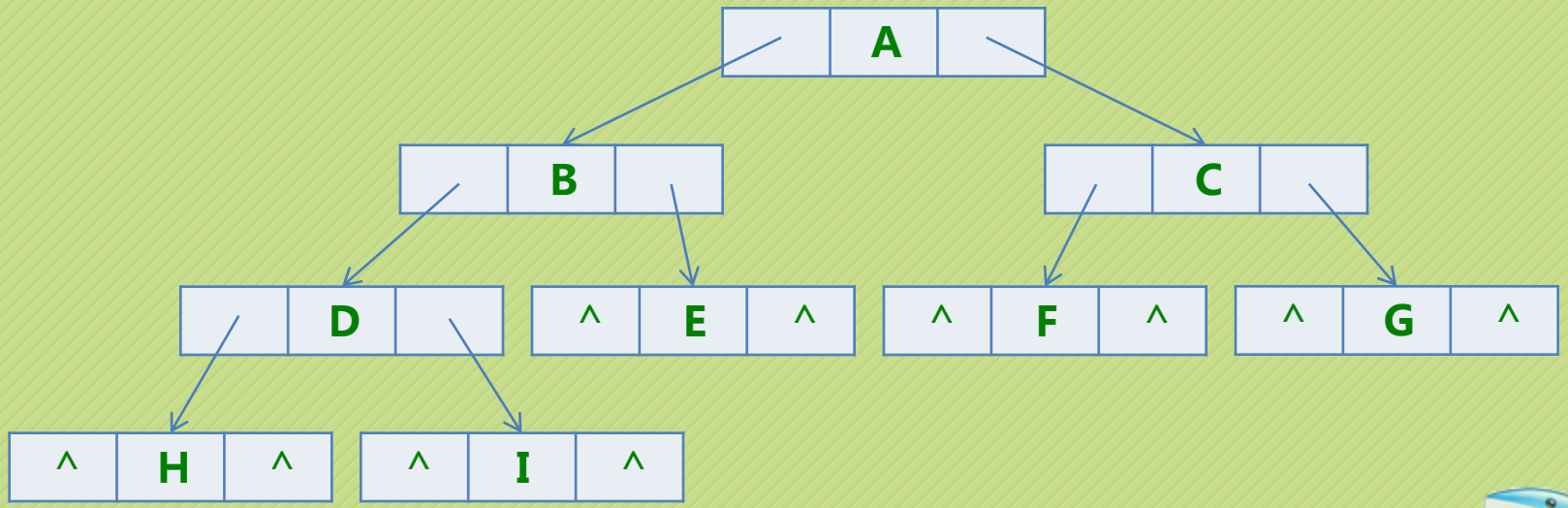
  一、浪费空间

  二、浪费时间

  三、浪费青春
- 来，我们具体分析下如何个浪费法：

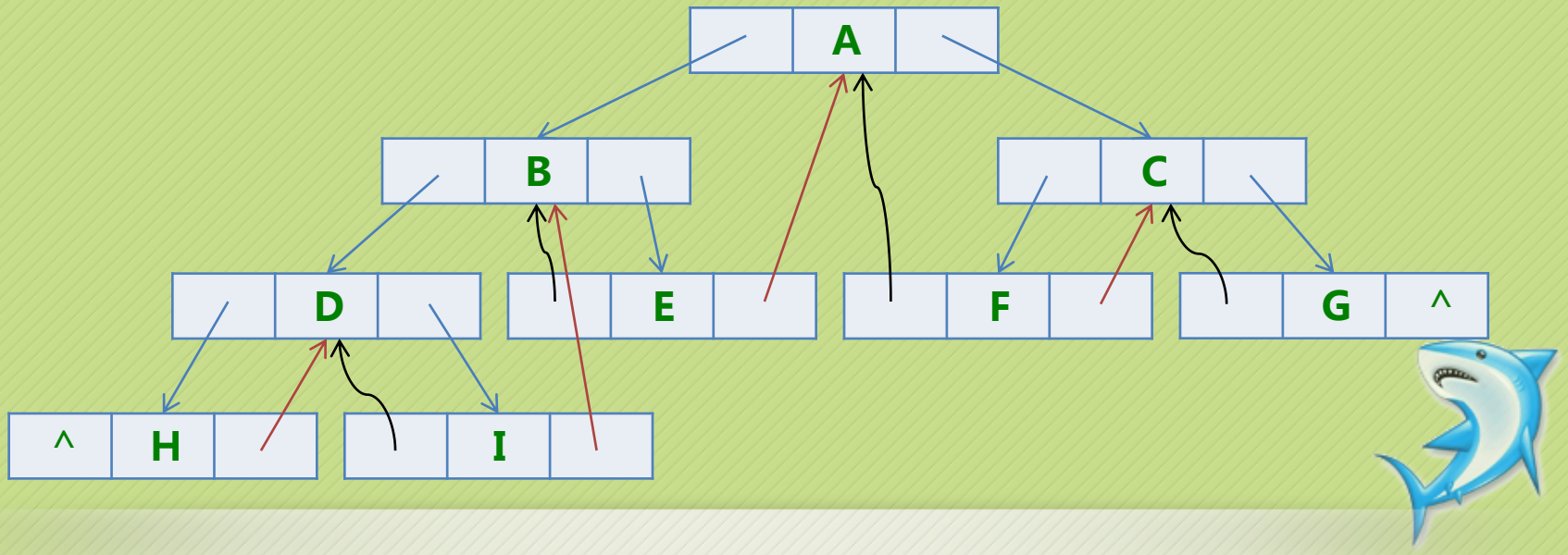
线索二叉树

- 数序题：请问以下有多少个“^”？总共浪费了多少字节的空间？(32bit的机器)



线索二叉树

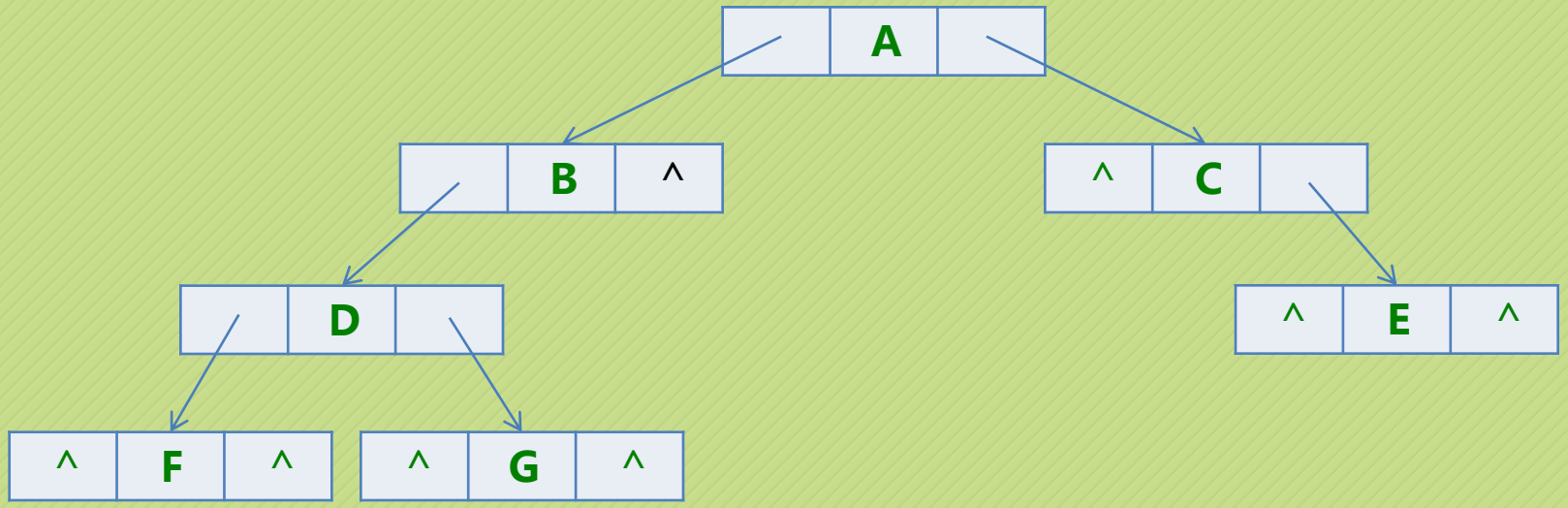
- 脑筋急转弯：我们知道通过对二叉树的约定遍历方式，可以得到一个固定的遍历顺序，那么请问哪种遍历方式可以节省“^”所浪费的空间？（利用“^”记录该结点的前驱后继）



线索二叉树

- 没错，中序遍历可以拯救地球！
- 上图经过中序遍历后结果是：HDIBEAFCG
- 我们发现红色的结点都是刚才“^”造成浪费的结点，利用中序遍历刚好它们均处于字符中间，可以很好地利用“^”来存放前驱和后继的指针。
- 不过好事总是多磨的，我们是有主观意识所以可以很容易出来哪些结点可以利用存放前驱后继。
- 有鱼油不同意了：这所谓的主观意识还不简单，不就是从第一个开始每隔一个结点都可以？

线索二叉树



- 上图经过中序遍历后结果是：FDGBACE
- 黄色说明只有一个空闲的指针位置，如果是这样的话我们就面临一个问题：机器怎么识别到底是存放指针还是线索？

线索二叉树

- 没错，她需要一丁点儿提示，为此我们将已经定义好的结构进行“扩容”：

lchild	ltag	data	rtag	rchild

  - ltag为0时指向该结点的左孩子，为1时指向该结点的前驱。
  - rtag为0时指向该结点的右孩子，为1时指向该结点的后继。
- 让我们一起来敲代码吧！