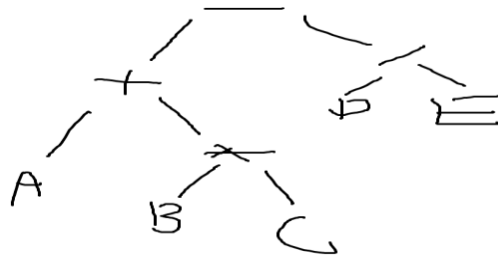


一、单项选择题

1. C
2. A
3. D
4. B
5. D
6. B

7. D 解析：正常题目无法通过后缀求前缀，但选择题不一样，只需观察选项是否满足条件，由后缀可得根节点为-，且根节点的右孩子为/，A为叶子节点，所以 A,B 选项很好排除，观察 C,D,可知 ABC 在根节点左边，DE 在右边，且可能都为叶节点，综上不难画出图形



8. 题目有误 解析：通过计算公式 $n(n+1)/2+1$, 得 11
9. A 解析： $n2 = n0-1$ ，且哈夫曼树没有度为 1 的结点
10. A 解析：快速排序在基本有序的时候算法时间复杂度是最坏的，此时为 $O(n^2)$
11. A
12. B
13. C 解析：取根号
14. C 解析：将第三位通过加 0.5 再强制转换达到四舍五入的效果
15. B 解析：B 选项 “Design” 后面有 `\0`，赋值给六个空间的字符数组，数据越界错误，注意 A 是每个字符单独赋值，是字符数组，但不是字符串
16. D 解析：第一个节点为头节点
17. C 解析：while 的循环体只有一条语句，是死循环
18. C
19. B
20. D 解析：静态变量只初始化一次，且函数被调用后值依旧保存，第二次调用 fun 函数时 static 的 a 值为 5
21. B
22. A 解析：宏替换后 $m+n*m+n*k = 1+2*1+2*3 = 9$
23. A
24. D
25. B 解析：枚举常量默认从零开始，并依次加 1，被赋值过后，从赋值处开始加 1
26. C 解析：strcpy(s1, s2)，是将 s2 赋值给 s1，并返回 s1

二、填空题

1. 操作

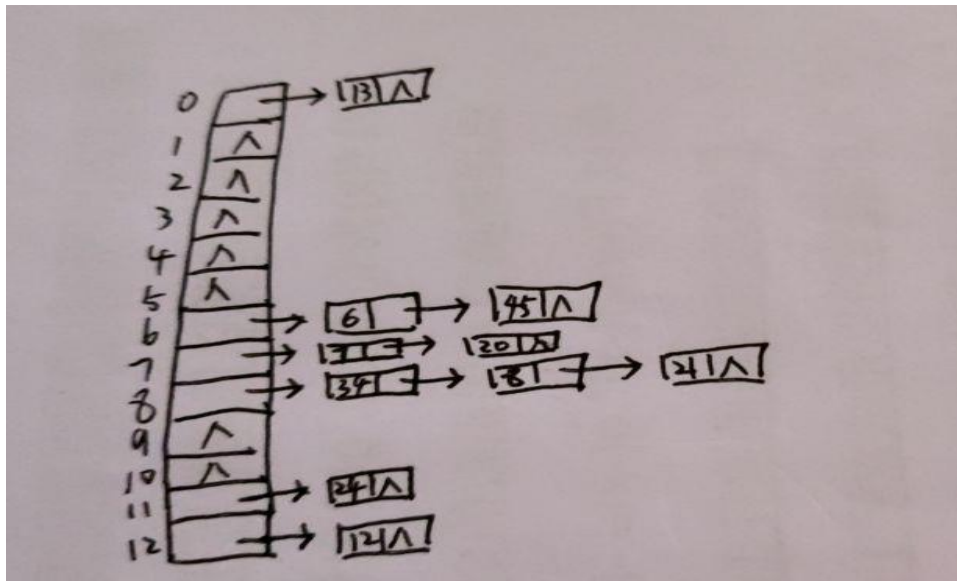
2. 5100 解析：s 的值为 $0+1+\dots+50+100+99+\dots+50$, 结果为 5100
3. 51 解析：完全二叉树度为 1 的结点，不超过 1 个，又 $n2 = n0-1$

4. 5 解析: 公有 $2n! / [(n+1)! * n!]$ 种可能, 即 $6! / 4! / 3! = 5$
5. 4 解析: 有公式 $\lfloor \log_2 n \rfloor + 1$
6. (b,c)
7. 12 解析: 第一行 1 个, 第二行 2 个, 第三行 3 个, 第四行 4 个, 第五行 3 个, 共 13 个对于下表 12
8. n-1
9. l<=n 解析: 通过 if 判断是否到底 i 的对应位置, 没到往下 next
10. p->next
11. p=p->next
12. 0
13. bt->data>'A'&&bt->data<='Z'
14. bt->data 解析: 注意 for 循环没有循环体, 只起一个遍历搜索作用
15. bt->rchild,s
16. #define MAX 10
17. i=0
18. index = j 解析: index 记录最小值下表, 并与每趟的第一个元素也就是 i 进行交换
19. t=a[i]
20. l+1 解析: for 循环起移动数组元素的作用, n 代表数组删除后的长度
21. *(a+j)
22. n--
23. i%4==0 解析: 注意 i 初始为 1
24. 6: count=max=0
25. 9: t=num%10 解析: 取最后一位, 看是否为 0, 是 count++
26. 16: num!=0
27. 5: sum=0
28. 6:for(n=50,n<=100,n++) 解析: 题目给出包扩 100
29. 8: if(n%3==0)
30. 11: printf("%ld\n",sum); 解析: 题目虽然有些模糊, 不难看出 sum 前面有个符号

三、简答题

1.

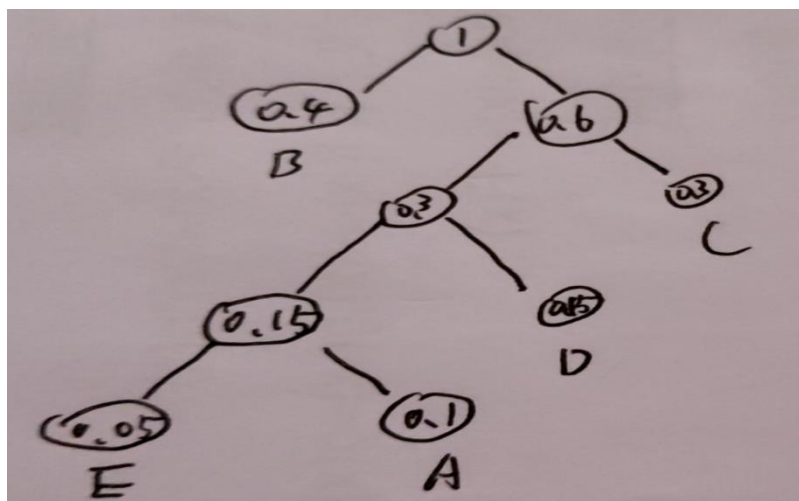
(1) hash 表如图:



(2) $ASL_{成功} = (1 + 1 + 2 + 1 + 2 + 1 + 2 + 3 + 1 + 1) / 10 = 1.5$

2.

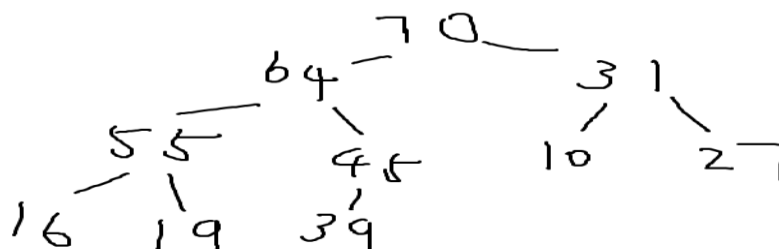
(1) 哈夫曼树如图:



(2) 平均编码长度: $[0.4 \times 1 + 0.3 \times 2 + 0.15 \times 3 + (0.05 + 0.1) \times 4] / 5 = 0.41$

3.

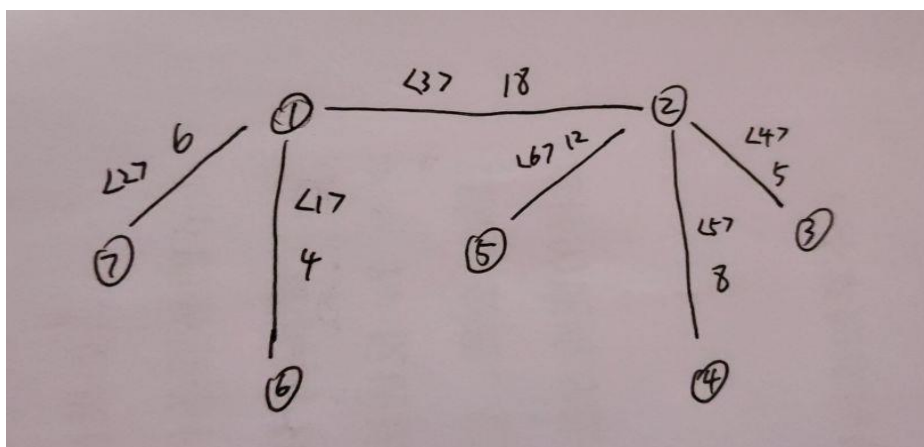
(1) 大根堆如图:



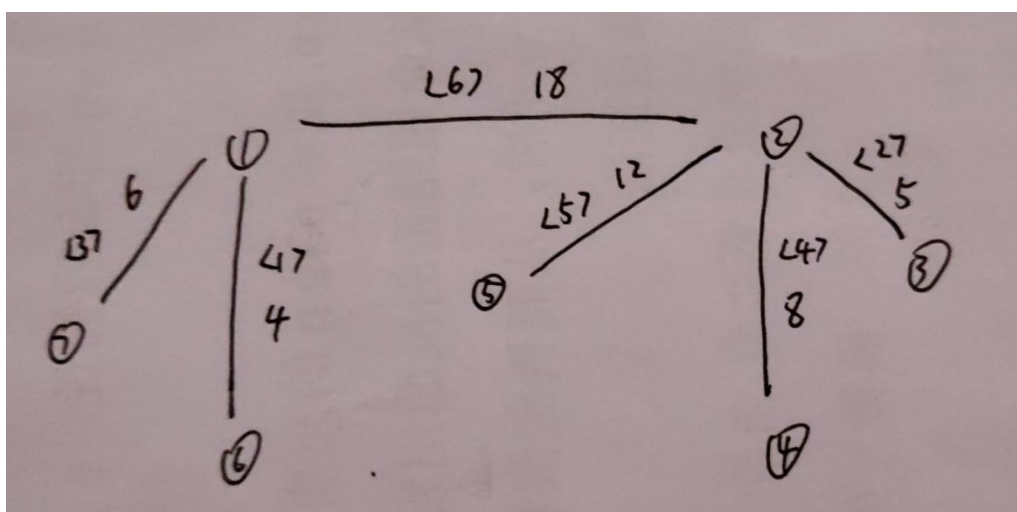
(2) 10 19 16 27 45 31 70 64 55 39

4.

Prim 算法:



Kruskal 算法:



5.4264 解析: 题目不太清晰, 但是只要注意递归就好了, 递归出口为 $n=1$, 且不难看出函数的作用为逆序输出

6. $a=2, b=2$

7. ABCDBCDCDD 解析: `strupr` 作用是变大写, `%s` 输出遇到 `\0` 才结束

8. 解析: 题目有些不太清楚, 设 `fun` 函数里面的是 $y=x$, 函数里面的 y 是局部变量, 所以几次调用 `fun` 函数, 结果互不影响

$y=-1$

$y=-2$

$y=-3$

9. 1 4 2 5 3 6

10. 程序不全, 难以求解, 基本是考察递归函数

四、算法及程序设计题

1.

```

int count(BT bt, char ch)//这里的 BT 是指针类型即传入根节点地址
{
    int sum=0;
    if(bt){
        if(b->data==ch)    sum++;
        sum+=count(b->lchild);
        sum+=count(b->rchild);
    }
    return sum;
}

```

2.

```

#include <stdio.h>
#include <string.h>
#define N 5      //学生数
struct STREC{
    char id[20]; //学号
    int score; //成绩
};
int fun(STREC *a, STREC *b)
{
    float max=0;
    int i,count=0; //count 记录最高人数
    for(i=0;i<N;i++){
        if(a[i].score>max){ //找出最高分
            max = a[i].score;
        }
    }
    for(i=0;i<N;i++){
        if(a[i].score == max){ // 将最高分的信息存入 b 数组
            strcpy(b[count].id, a[i].id);
            b[count].score = a[i].score;
            count++;
        }
    }
    return count;
}
void PrintInfo(STREC *t, int num)
{
    int i;
    printf("分数最高的学生信息如下（学号，分数）:\n");
    for(i=0;i<num;i++){
        printf("%s %d\n",t[i].id,t[i].score);
    }
}

```

```
Void main()
{
    struct STREC s[N]={{"2019001", 75}, {"2019002", 85}, {"2019003", 85}, {"2019004",
85}, {"2019005", 80}};
    struct STREC t[N];
    int num;
    num = fun(s,t);
    PrintInfo(t, num); //输出函数
    return 0;
}
```