武汉大学数学与统计学院2017-2018学年第一学期期末考试
数据结构与算法（B卷）

姓名:_____ 学号:_____

1. （20分） Python相关

   (a) （5分）编写一段代码，用while循环计算100以内所有奇数之和。

   (b) （5分）仔细阅读以下程序，写出运行结果：

```
alist = [1, 3, 5, 7, 9]
print(alist[3:5])
print(alist[:4:2])
print(alist[-3:])
print('ABCDEFG'[:3])
print('ABCDEFG'[::2])
```

   (c) （5分）用列表生成式(list comprehension)将下面列表中所有的字符串变成小写:

```
list = ['Hello', 'World', 'IBM', 'Apple']
```

   (d) （5分）给半径（ratio），编写圆（Circle）的类，其中包括求面积（area）和周长（circum）的方法。

2. （15分） 对于有序链表，设其结点类为

```
class Node(object):
    def __init__(self, data):
        self.data = data
        self.next = None
```

   补充以下代码，以实现有序链表类：

```
class OrderedList:
    def __init__(self):
        self.head = None

    def search(self,item):
        current =self.head
        found = False
        stop = False
        while ...:
            if current.data == item:
                found = True
            else:
                if current.data > item:
                    stop = True
                else:
                    ...
        return found

    def add(self,item):
        ...
        previous = None
        stop = False
        while current != None and not stop:
            if current.data > item:
```

```
                stop = True
            else:
                previous = current
                current = current.next

        temp = Node(item)
        if previous == None:
            ...
            ...
        else:
            temp.next = current
            previous.next = temp
```

3. （15分）假设有以下栈类的定义

```
class Stack(object):
    def __init__(self):
        self.items = []
    def isEmpty(self):
        return self.items == []
    def push(self, item):  //入栈
        ***
    def pop(self):         //出栈
        ***
```

试补充以下函数，实现后缀表达式的求值：

```
def postfixEval(postfixExpr):
    stack = Stack()
    tokenList = postfixExpr.split()

    for token in tokenList:
        if token in "0123456789":
            ...
        else:
            ...
            ...
            ...
            ...
    return stack.pop()

def doMath(op, op1, op2):
    if op == "*":
        return op1 * op2
    elif op == "/":
        return op1 / op2
    elif op == "+":
        return op1 + op2
    else:
        return op1 - op2
```

4. （15分）

(a) （5分）编写完整程序实现二叉树的后序遍历：

```
class BinaryTree(object):
    def __init__(self, data):
        self.data = data
        self.lchild = None
        self.rchild = None

    def postorder(self):
        ...
```

(b) （10分）编写完整程序，往二叉查找树中插入节点：

```
class BinarySearchTree(object):
    def __init__(self, data):
        self.data = data
        self.lchild = None
        self.rchild= None

    def insert(self, data):
        ...
```

5. （20分）

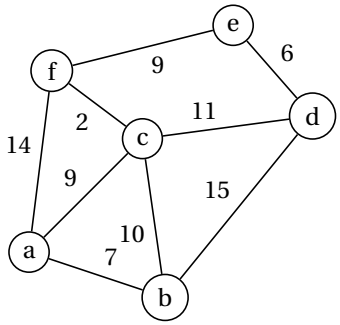
(a) （10分）写出完整程序，实现有序列表的折半查找：

```
def binarySearch(alist, item):
    found = False
    ...
    return found
```

(b) （10分）写出完整程序，实现列表的选择排序：

```
def selectionSort(alist):
    ...
```

6. （15分）给定无向图



(a) （5分）写出邻接矩阵；

(b) （4分）以字典的形式，写出每个顶点的度；

(c) （6分）根据该邻接矩阵，写出从顶点 a 出发，深度优先和广度优先搜索的结果。