中原大學 105 學年度 ☑上學期 考試題卷 ☑期中

92

科目名稱：資料結構　　開課班級：資工二甲/資工二乙　　油印份數：72/74

考試時間：11 月 10 日 第 4 節　　科目代號：CS103D/CS103E　　教師姓名：吳宜鴻

☑不可使用計算機、翻譯機或手機　☑可以使用紙本字典　本份試題共 4 頁，本版面為第 1 頁

☑直接在命題紙上作答（背面可作為計算用紙）　☑評分將不予部分給分，作答務必力求完整

系級：資訊二乙　學號：10427248　姓名：陳其彥

I. Single-Choice Problems (50%) 每題 3 分，共 20 題，答錯一題倒扣 1 分，滿分以 50 分為上限　+50

A 1. In the recursive solution to _____, the problem size is decreased by half at each recursive call. (A) finding the maximum (B) Towers of Hanoi (C) finding the *k*-th smallest (D) calculating the factorial

C 2. Which of the following statements is FALSE about a *constructor* in C++? (A) a constructor has no return type (B) a constructor may have a parameter (C) a class can have only one constructor (D) a constructor has the same name as the class

B 3. Which of the following is TRUE about *doubly linked lists*? (A) the last node points to the first node (B) each node has both a precede pointer and a next pointer (C) the precede pointer of the last node has the value NULL (D) the precede pointer of the first node references the last node

D 4. Which of the following is NOT in the *language* S defined by the following *grammar*? (A) 0a1 (B) 00b1 (C) 00a11 (D) 00b11

<S> = *a* | 0<Y>

<Y> = *b* | <S>1

B 5. The *midpoint* of a sorted array has the index _____, where *x* is the index of the first item in the array, and *y* is the index of the last item in the array. (A) $x/2 + y/2$ (B) $x + (y - x)/2$ (C) $(y - x)/2$ (D) $(x - y)/2$

A 6. Which of the following is the benefit of *modularity*? (A) it isolates errors (B) it runs faster (C) it is easier to make a copy (D) it saves more space

A 7. In a *circular linked list*, _____. (A) the last node points to the first node (B) the next pointer of the first node has the value NULL (C) the next pointer of each node has the value NULL (D) the precede pointer of the dummy head node references the last node

C 8. If a *stack* is used to check balanced braces, the method _____ is invoked when "{" is reached. (A) isEmpty (B) getTop (C) push (D) pop

D 9. What happens if a recursive function never reaches a *base case*? (A) the function returns the correct value (B) the function returns an incorrect value (C) the function terminates immediately (D) an infinite sequence of recursive calls occurs

A 10. In the recursive solution to the *Eight Queens* problem, the problem size decreases by ___ at each recursive step. (A) one column (B) two columns (C) one square (D) two squares

科目名稱：資料結構　　開課班級：資工二甲/資工二乙

考試時間：11 月 10 日 第 4 節　　科目代號：CS103D/CS103E

A 11. How many data movements (i.e., shifting the items) will the following algorithm perform on a list of 10 items? (A) 126 (B) 135 (C) 40 (D) 145

```
reverseList (in aList:List, out success:boolean)
 for (i = 1 to aList.getLength() - 1)
 {   aList.retrieve(1, dataItem, success);
     aList.insert(aList.getLength() – i + 2, dataItem, success);
     aList.remove(1, success);                                    }
```

A 12. Given the declaration: int x = 1, *p = new int; which of the following statements will cause a *memory leak*? (A) p = &x; (B) *p = x; (C) delete p; (D) *p = -1;

B 13. Based on the recursive definition, compute the return value of ***Acker***(1, 2). (A) 5 (B) 4 (C) 3 (D) 2

Acker(m, n) = n + 1, if m = 0
= ***Acker***(m-1, 1), if n = 0
= ***Acker***(m-1, ***Acker***(m, n-1)), otherwise

D 14. The following way to define two methods with the same name in a class is called _____. (A) inheritance (B) exception (C) overriding (D) overloading

```
class Rational {    …   public: …
                        Rational add(Rational);
                        Rational add(long);      }
Rational Rational::add(Rational r) {          …          }
Rational Rational::add(long i)     {          …          }
```

B 15. Which of the following is impossible for a *doubly linked list* with a *dummy head*? (A) each node has both a precede pointer and a next pointer (B) the precede pointer of the last node has the value NULL (C) the last node points to the dummy head (D) the precede pointer of the first node references the dummy head

C 16. Which of the following is NOT a valid *postfix expression*? (A) a b - c d + - (B) a b * c + d * (C) a b c - d * (D) a b c + /

C 17. Among the following statements about *recursive binary search*, which one is TRUE? (A) it can be applied to an unsorted array (B) it starts by comparing with the first item in the array (C) it has two base cases (D) it searches exactly one half of the array

C 18. By default, all members in a C++ *class* are _____. (A) public (B) protected (C) private (D) NULL

D 19. Which is one of the required steps to delete a node from a *linked list*? (A) retrieve the data from a node (B) connect a node to the linked list (C) release the space of a node (D) allocate the space of a node

A 20. What is the value of the *prefix expression*: * + 8 * 4 - 5 3 2 ? (A) 32 (B) 10 (C) 60 (D) 68

科目名稱：資料結構　　開課班級：資工二甲/資工二乙

考試時間：11 月 10 日 第 4 節　　科目代號：CS103D/CS103E

II. Simple-Answering Problems (30%) 每格 3 分，共 11 格，作答完整才得分，滿分以 30 分為上限

1. Complete the pseudo codes in terms of the ADT **List** operations.

(i) Exchange the first element and the last element in a list.

```
void swapFirstLast(in aList, out success)
    aList.retrieve(1, firstItem, success);
    aList.retrieve(aList.getLength(), lastItem, success);
    aList.(1) remove (1, success);
    aList.remove( (2) aList.getLength(), success);
    aList.insert(1, lastItem, success);
    aList.insert( (3) aList.getLength()+1, firstItem, success);
```

Operation Contract for the ADT **List**
createList()
destroyList()
isEmpty():boolean
getLength():integer
insert(**in** index, **in** newItem, **out** success)
remove(**in** index, **out** success)
retrieve(**in** index, **out** dataItem, **out** success)

(ii) Reverse the order of all elements in a list.

```
void reverseElementOrder(in aList, out success)
for (int i = 1; i < (4) aList.getLength() ; ++i)
{   aList.retrieve(1, dataItem, success);
    aList.remove(1, success);
    aList.insert( (5) aList.getLength()+1 , dataItem, success);
}   // end for
```

2. Consider the language **S** as defined by the following *grammar* to answer the following questions.

<S> = <U> | <S> <D>
<D> = 1 | 0
<U> = A | B

(i) Write all strings in the language that have exactly three characters.

Answer: (6) A11, A10, A01, A00, B11, B10, B01, B00

(ii) Is the string AB0011 included in this language?

Answer: (7) No

(iii) Modify the above grammar to define a language of bit-strings that the first character must be 1 and the last character must be 0.

Answer:

(8) <S> = <D><U> | <S><U>
<D> = 1 | <D>
<U> = <U> | 0

3. Given the infix expression ((a + (b / (c – (d + e) * (f – g)))) – h), answer the following questions about evaluating it by using a stack.

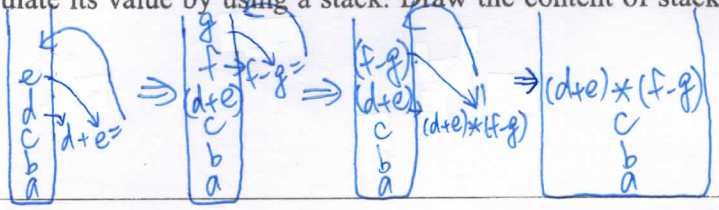
(i) The first step transforms it into the postfix expression:

Postfix expression: (9) abcde+fg-*-/+h-

(ii) The second step is to calculate its value by using a stack. Draw the content of stack after the operator '*' has been calculated.

Content of stack:

(10)



(iii) The above way to evaluate postfix expression can also be applied to prefix expression. Explain how.

Answer:

(11) -+a/b-c*+de-fgh，運算符號在數字左邊

科目名稱：資料結構 +1f 開課班級：資工二甲/資工二乙

考試時間：11 月 10 日 第 4 節 科目代號：CS103D/CS103E

III. Advanced Problems (20%)每一空格 3 分，共 7 空格，作答完整才得分，滿分以 20 分為上限

1. Give short C++ codes as an example to explain the meaning of each term. ***Get no point if there is no code!***

- **encapsulation**:

```
class ex {
    ⋮
(1) };
```

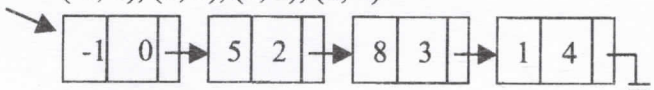
，封裝為將程式碼隱藏起來又讓使用者使用其中function.

- **inheritance**:

```
class ex : public exsuper
{
    ⋮
(2) };
```

，ex為繼承 exsuper這個父類別的子類別

2. Any polynomial can be organized as a linked list in a way that a non-zero term is stored on a node. For example, $x^4 + 8x^3 + 5x^2 - 1$ has four terms: (-1, 0), (5, 2), (8, 3), (1, 4).




```
struct Node
{   double   c;   //coefficient
    int      p;   //power
    Node     *next;
}   // end Node
```

Each node has three fields: c, p, and next. Fill in the blanks to complete the following functions.

(a) Make a copy of the given linked list.

```
Node *copyList(Node *oldList)
Node *newList, *newPtr, *oldPtr;

if (oldList == NULL)
    newList = NULL;  //the original list is empty
else {
    newList = new Node;    //the first node
    newList->c = oldList->c;
    newList->p = oldList->p;
    newPtr = newList;
    oldPtr = oldList->next;  //the second node
    while (oldPtr != NULL)
    {   (3) newPtr->next = new Node;
        newPtr = newPtr->next;
        newPtr->c = oldPtr->c;
        newPtr->p = oldPtr->p;
        oldPtr = oldPtr->next;
    }   //end while
    (4) newPtr = NULL ;  //the tail
    return newList;
}   //end else
```

(b) Add two polynomials.

```
Node *addPoly(Node *polyX, Node *polyY)
Node *polyZ = NULL, *dummy = NULL;

if (polyX == NULL)          //only Y
    polyZ = copyList(polyY);
else if (polyY == NULL)     //only X
    polyZ = copyList(polyX);
else {
    dummy = new Node;       //a dummy head
    polyZ = dummy;
    do { (5) polyZ->next = new Node;
        polyZ = polyZ->next;   //create a new term
        if (polyX->p <= polyY->p)
        {   polyZ->p = polyX->p;
            if (polyX->p == polyY->p)
            {   polyZ->c = polyX->c + polyY->c;
                polyY = polyY->next;   //move Y
            } else   polyZ->c = polyX->c;
            polyX = polyX-> next;      //move X
        } else {
            polyZ->p = polyY->p;
            polyZ->c = polyY->c;
            (6) polyY = polyY->next;   //move Y
        }   // end if-else
    } while (polyX!=NULL) && (polyY!=NULL);
    if (polyX == NULL)          //remaining Y
        polyZ->next = copyList(polyY);
    else if (polyY == NULL)     //remaining X
        polyZ->next = copyList(polyX);
    (7) polyZ = polyZ->next ;
    dummy->next = NULL;
    delete dummy;               //remove the dummy head
}   // end else
return polyZ;
```