淡江大學九十三學年度碩士班招生考試試題

系別：資訊工程學系　　科目：計算機概論(含資料結構、程式語言結構)

准帶項目請打「○」否則打「x」
簡單型計算機

本試題共 2 頁

本試題雙面印製

1. Explain what fixed-point numbers are and the relationship between integers and fixed-point numbers. (10 pts.)

2. State the purpose of ***program counter*** in computer architectures and explain how it is updated in instruction execution cycle. (10 pts.)

3. Consider the function `mysterious` in C code given in the right box:

   a. State the purpose of the function. (5 pts.)

   b. What is a possible problem may be encountered in this function? (5 pts.)

```
int mysterious(int n) {
    while( n>1 ) {
        if( n%2 == 0 )
            n/=2;
        else
            n=3*n+1;
    }
    return n;
}
```

4. What is wrong with the following function in C ? (5 pts.)

```
int*  f(void)  { int x = 0; return &x; }
```

5. Give an example to show that the following macro definition in C which defines the square of input expression `x` is a bad (incorrect) design. (5 pts.)

```
#define   sq(x)  x * x
```

6. Consider the following statements in and C and C++ both.

   (i) `++(i++);` (ii) `(++i)++;`

   a. Which one of the above statements is illegal in both C and C++? (5 pts.)

   b. Use l-value and r-value concepts to explain why it is illegal. (5 pts.)

7. Consider the traversal *pre-*, *post-* and *in*-orders of binary trees. Let u and v be two nodes of a binary tree T. Which of the following statements are true if u is a proper ancestor of v. Justify your answers. (15 pts.)

   (a) *pre*-order(u) < *pre*-order(v)

   (b) *in*-order(u) < *in*-order(v)

   (c) *post*-order(u) < *post*-order(v)

8. Write a **recursive** C function which returns the square (x^2) of an input integer x **without** using multiplications (*). (hint : consider the trivial equation $x = (x\text{-}1) + 1$) (15 pts.)

◀注意背面尚有試題▶

淡江大學九十三學年度碩士班招生考試試題

系別：資訊工程學系　　科目：計算機概論(含資料結構、程式語言結構)

准帶項目請打「○」否則打「×」
簡單型計算機

9. Consider the following parentheses parsing program in C. Give the procedure activation (calling) tree for input string (< < >) { >) (10pts.)

```
char inch;
void mat(char s) {
    if( inch != s ) fprintf(stderr,"error: inch");
    inch = getchar();
} // end of mat

void  bexp(void) {
     while(1) {
        switch(inch) {
        case '(' : mat('('); bexp(); mat(')'); continue;
        case '}' : mat('}'); bexp(); mat('{'); continue;
        case '<' : mat('<'); bexp(); mat('>'); continue;
        default: return;
        } // end of switch
    } // end of while
} //end of bexp

int main(void) {
    inch = getchar();
    bexp();
    return 0;
} // end of main
```

10. The following procedure ***DELETE_ALL*** was intended to remove all occurrences of element x from list L. Explain why it does not work and suggest a way to repair the procedure so it performs its intended job. Assume that functions ***FIRST*** and ***LAST*** return the addresses of the first and the last elements of list L, respectively, and function ***DELETE*** deletes the element whose position is stored in p->next. (you can use the line number given below in your answer) (10 pts.)

```
1:  void  DELETE_ALL( dataType x, LIST L) {
2:        node *p;
3:        p = FIRST(L);
4:        while(  p != LAST(L)  ) {
5:             if(  p->element == x  )
6:                  DELETE(p,L);
7:             p = p->next;
8:        } /* end of while */
9:  }
```