# 國立中央大學　資訊工程學系
# 111學年度 碩士在職專班 招生入學考試命題紙

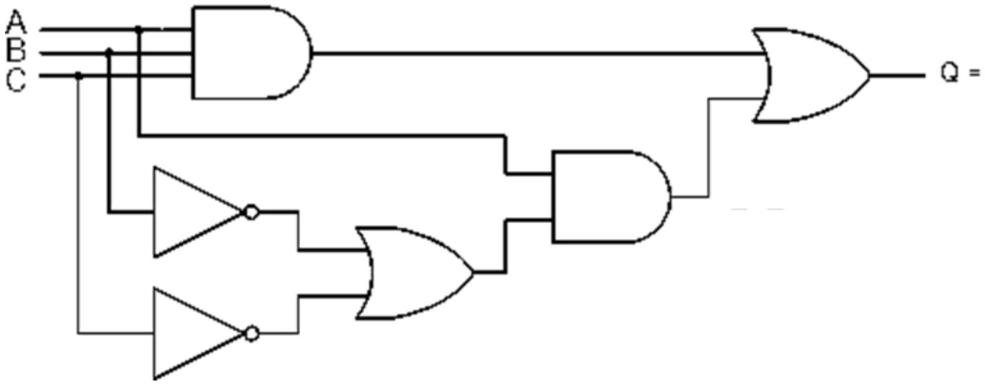
第1-7題每題4分、第8題8分、第9題6分、第10題8分

1. The memory address of last location of a 64K byte memory chip is FFFFFH. Find the starting address.
   (A) EFFFFH (B) F0000H (C) E0000H (D) F0001H

2. In which cycle the memory is read and the contents of memory at the address contained in the Program Counter (PC) register are loaded into Instruction Register (IR).
   (A) Execution (B) Memory (C) Fetch (D) Decode

3. Given the equation: $(1001101.0010101)_2 - (D.AA)_{16}$. What is the result of the equation?
   (A) $(62.5)_{10}$ (B) $(64.5)_{10}$ (C) $(63.125)_{10}$ (D) $(63.5)_{10}$

4. Select the correct boolean expression for the following logic circuit.
   (A) Q = (A.B) + ($\overline{A+B}$)　　(B) Q = (A+B) . ($\overline{A+B}$)
   (C) Q = (A+B) + ($\overline{A.B}$)　　(D) Q = (A+B) . ($\overline{A.B}$)



5. In a 32-bit addressing architecture, a cache consists of multiple 32-byte blocks. Suppose that the cache can store 256K bytes. How many blocks can be stored in the cache?
   (A) 1024 (B) 2048 (C) 4096 (D) 8192

6. Which statement is correct?
   (A)RAM is a read-only device. (B) ROM can be read and write.
   (C) RAM cannot be persistent storage because of its volatility.
   (D) ROM cannot persistently preserve program.

7. What is an inclusive cache? Give one example to explain the inclusion property for a multi-level cache hierarchies. Additionally, please list one advantage and disadvantage of the inclusive cache design.

8. Assume that 32-bit memory address format is adopted to a 2-way set-associative cache. The 32-bit memory address information can be divided into 10 bits of index, 18bits of tag, and 4 bits of offset. In this question, we start with an empty cache. Show the tag, index, and way information for each memory reference as follow. Additionally, you need to indicate the reference result (i.e., hit or miss) for each memory access. If it is a cache miss, show what type of cache miss it is in the 3-C model (without coherence). (**8 pts**)

| Address | Tag | Index | Way | Hit/Miss (Types) |
|---|---|---|---|---|
| **0x00773F11** | | | | |
| **0x00553F12** | | | | |

9. If the floating-point (FP) instructions of an application on a specific processor ***C1*** consumes 60% of the total execution time. Moreover, in the same application, 25% of the floating-point time is taken to do square root calculations.
   - A new processor ***C2*** is developed. This new processor could either enhance the performance of FP instructions by a factor of 1.5 or alternatively increase the performance of the square root operation by a factor of 8. Which is the better design for the aforementioned application? (**6 pts**)

10. Assume one byte data value is 00111011two.
    - First show the Hamming ECC code for that byte (**4 pts**)
    - When bit 5 is flipped, please show that the ECC code finds and corrects the single bit error. (**4 pts**)

第 11-23 題每題 4 分

11. The output of the following code is (A) 33, 39 (B) 33, 14 (C) 99, 18 (D) 99, 26

```
#include <iostream>
#include <queue>
#include <stack>
using namespace std;
int main(){
    //que and stk are declared to be a queue and a stack, respectively
     int ans1, ans2;
     queue<int> que; stack<int> stk;
     que.push(6); stk.push(2); que.push(2);
     stk.pop(); que.push(9); que.pop();
     stk.push(7); que.pop(); que.push(5); stk.push(11);
     ans1= que.front()*stk.top();
     que.push(2); stk.pop(); stk.push(8);
     ans2 = que.front()*que.back();
     cout<<ans1<<", "<<ans2<<endl;
    return 0;
}
```

**Consider the pseudo code of circular queue below and answer Question 12 to 14.**

```
#define MAX_SIZE 5
struct queue{
    int   full;
    int front,rear;
    int data[MAX_SIZE];
};
void Setqueue(struct queue *que);
void Addq(struct queue *que, int x);
int Deleteq(struct queue *que);
int IsFull(struct queue *que);
int IsEmpty(struct queue *que);
void Setqueue(struct queue *que){
    que->full=0; que->front=0; que->rear=-1;
    return;
}
```

```
void Addq(struct queue *que, int x){
    if(IsFull(que)){
        printf("Queue is full\n");
        return;
    }
    que->rear = (que->rear + 1) % MAX_SIZE;
    que->data[que->rear] = x;
    if((que->rear+1)%MAX_SIZE ==_①_) que->full =1;
    return;
}
int Deleteq(struct queue *que){
    int value;
    if (IsEmpty(que)){
        printf("Queue is empty\n");
        return -1;
    }
    que->full = 0;
    value = que->data[_②_];
    que->front%=MAX_SIZE;
    return value;
}
int IsFull(struct queue *que){
    return (que->full==1)?1:0;
}
int IsEmpty(struct queue *que){
return ((que->full==0)&&((que->rear+1)%MAX_SIZE)==que->front )?_③_;
}
```

12. Blank ① should be (A) que->rear (B) que->full (C) que->front (D) 0

13. Blank ② should be (A) (que->front) (B) (que->front)++ (C) ++(que->front) (D) (que->front)--

14. Blank ③ should be (A) 1:0 (B) 0:1 (C) 2:3 (D) 3:2

15. What does the following function do in general? (A) Prints binary representation of n in reverse order. (B) Prints binary representation of n. (C) Prints the value of (log n). (D) Prints the value of (log n) in reverse order.

```
void fun(int n){
      Stack S;    //Say it creates an empty stack S
      while(n>0){
            push(&S, n%2);
            n = n/2;
      }
      //Run while Stack S is not empty.
      while(!isEmpty(&S))
      printf("%d", pop(&S));
}
```

16. Consider the pseudo code below. Which is the output when input is "NCUCSIE"?
(A) NCUCSIENCUCSIE
(B) EISCUCN
(C) NCUCSIE
(D) EISCUCNEISCUCN

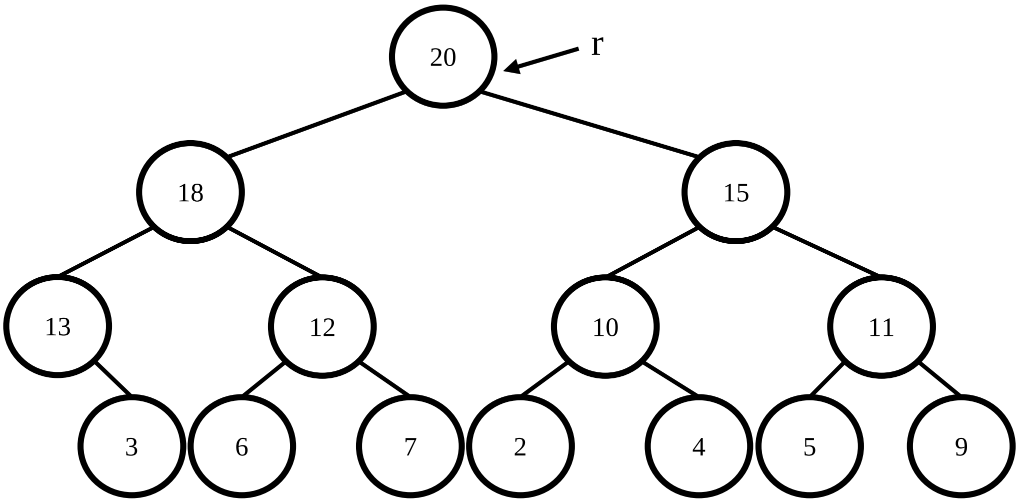
```
while (there are more characters in the word to read){
      read a character
      push the character on the stack
}
while (the stack is not empty){
      pop a character off the stack
      write the character to the screen
}
```

**Consider the following pseudo code and answer Question 17 to 20.**

```
// struct node represent binary trees
  struct node{
          int data;
          struct node *left, *right;
  }
  int n=0;
  void F(struct node *p){
        if (p == NULL) return;
        if (n==0){
                  printf(“%d ”, p->data);
                  n++;
          }
        if (p->left != NULL)
                  printf(“%d ”, p->left->data);
          if (p->right != NULL)
                  printf(“%d ”, p->right->data);
          F(p->left);
          F(p->right);
  }
```

17. Function F() traverses a tree with
    (A) preorder
    (B) inorder
    (C) postorder
    (D) level order

18. Consider the following binary tree. What is the last output of F(r)? (A) 20 (B) 6 (C) 9 (D) 2



19. Function F has time complexity (A) O(log n) (B) O(n) (C) O(nlog n) (D) O($n^2$)

20. Consider a binary search tree which stores distinct elements. The keys in the left subtree of a node are smaller than those in the right subtree of the node (if any). Denote *NIL* as a null pointer. Pointer *p* and *left* direct to a node's parent node and the left child node, respectively. Function *Tree_ Maximum* (n) returns the maximum node (i.e., the node that has the biggest key) of a sub-tree rooted at node n. Assume that x is a node in the binary search tree. The following code fragment is executed. After that, y is not equal to *NIL*.

```
if left[x]≠NIL
        then y = Tree_ Maximum (left[x]); return y;
y =p[x]
while y≠NIL and x = left[y]
        do x = y and y = p[y]
return y;
```

Which of the following statements are (or is) true?
(A) y must be the parent node of x
(B) The key of y must be smaller than that of x.
(C) x must be the maximum node of a sub-tree rooted at node y.
(D) y must be the minimum node of a sub-tree rooted at node x.

21. After inserting the following integers into an empty max heap, what is the leftmost node of the resultant max heap?

    40 6 15 60 80 75 10 7 50 45

    (A)15 (B) 6 (C) 80 (D) 20

22. What is the postfix expression of infix expression ((q*k)+2(m+4))/(p*(a-b)).

    (A) qk*2m4++pab-*/

    (B) qk*2m4*++pab*-/

    (C) qk*2m4+*+pab-*/

    (D) qk*2m4++*p*ab-/

23. Given a PUSH sequence and a POP sequence. Which of the following statements is not true? (A) PUSH 1 2 3; POP 1 2 3; It must be a Queue (B) PUSH 1 2 3; POP 3 2 1; It must be a Stack (C) PUSH 1; POP 1; It must be a Queue (D) PUSH 1 2 3; POP 2 3 1; It is neither Stack nor Queue