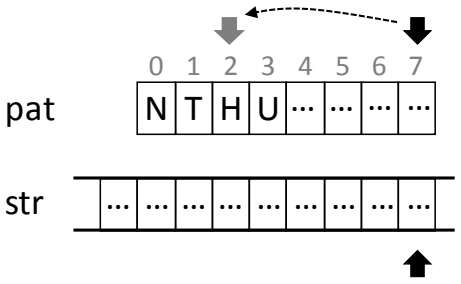
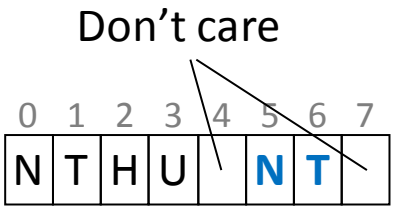
1	2	3	4	5	6	7	8	9	10

Data Structure Midterm Examination
3:30pm-5:20pm (110 minutes), April. 23, 2018

1. (8pt) KMP Algorithm

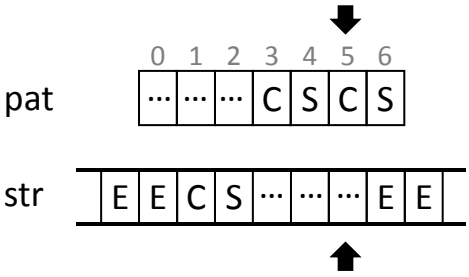
(1) Please show a pattern (NTHU…) that lets KMP Algorithm try to match pat[2] upon a mismatch at pat[7] as shown in the following figure.





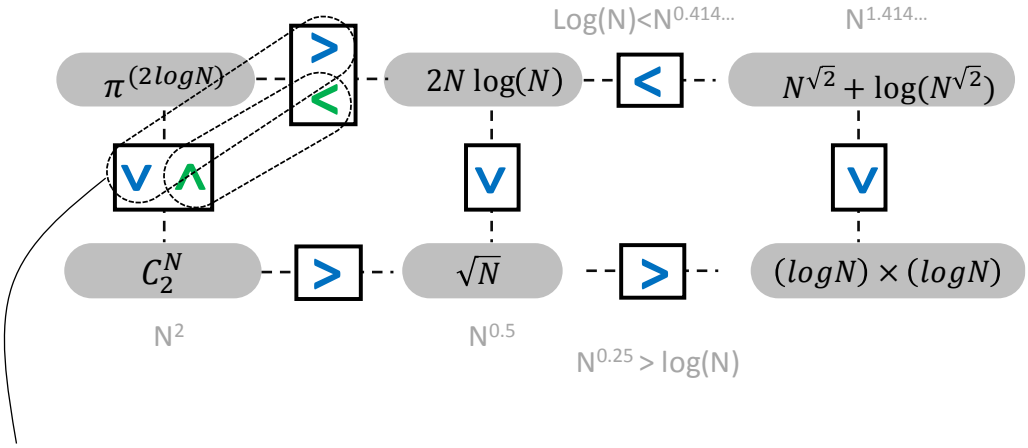
Ans: **pat** =

(2) What is the next index (0 ~ 6) KMP Algorithm would try to match upon a mismatch at pat[5] as shown in the following figure?



Ans: **next index** = **0**

2. **(14pt)** Please compare the asymptotic complexity hierarchy of the following functions using '=', '>', or '<'.



The answer should be either green or blue pairs because

$\pi^{(2log_2 N)} = N^{2log_2 \pi} > N^2 > Nlog(N)$

$\pi^{(2log_{10} N)} = N^{2log_{10} \pi} < N < Nlog(N) < N^2$

3. **(10pt)** Please complete the function that inserts a singly linked list (non-circular, with a header) into a **doubly linked list** (non-circular, with a header) right after the node pointed by the pointer P. Please assume the handling pointer of the doubly linked list is "first".

void DoublyList::InsetSinglyList(Node* List, Node* P)
{

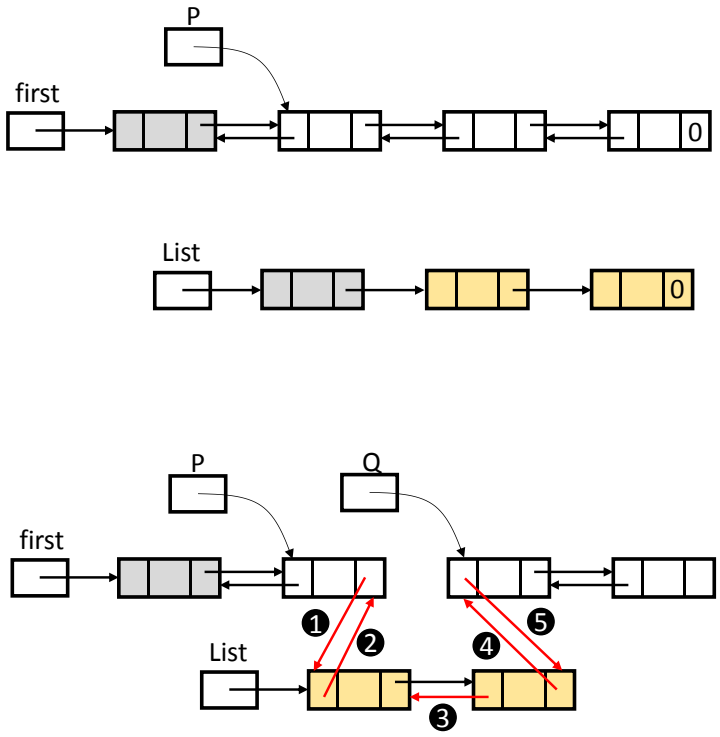
```
if (List->next == 0) return;
todel = List;
List = List-next;
delete todel;

if (first->next == 0){
  first->next = List;
}

Node * Q = P->next;
P->next = List; ❶
List->prev = P; ❷

for(; List->next!=0; List=List->next){
  List->next->prev = List; ❸
}

List->next = Q;    ❹
if (Q!=0){
  Q->prev = List; ❺
}
```



}

4. **(10pt)** Please analyze the **time complexity** of the following functions as tight as possible:

```
void f(int X, int Y, int a[])
{
   while (Y > 0) {
      if(X <= 0) {
         X = Y = Y - 1;
      }
      X = X - 1;
      a[Y]++;
   }
}
```

```
void g(int N, int a[], int b[])
{
   if (a[N] % N == 0)
      return;
   for (int i=1; i<N; i=i*2)
      b[N]++;
   g(N-1, a, b);
}
```

f(X, Y, …) ∈ Ω(Y^2 **)** **g(N, …) ∈ Ω(** 1 **)**

f(X, Y, …) ∈ O(Y^2 **)** **g(N, …) ∈ O(** log (N!) **) = O(** NlogN **)**

5. **(20pt)** Select the best answer (-1 points for each wrong selection)

__**A**__ Which can cause operating systems (e.g., Linux and Windows) to report a runtime error?

(A) Reference to an out-of-bound array entry.
(B) Miss a semicolon (分號) in a C program.
(C) Reference to a variable name that is undefined.
(D) Duplicate functions or variables in a program.
(E) None of the above.

__**D**__ Which corresponds to the slowest algorithm when the problem size is large enough?

(A) $\mathbf{O}(n^4)$
(B) $\mathbf{O}(0.1(4^n))$
(C) $\mathbf{O}(10\log(4^n))$
(D) $\mathbf{O}((4n)^n)$
(E) None of the above is the best answer because it depends on the implementation of the algorithm.

__**D**__ Which programming language can solve the largest set of problems.

(A) Structure language (e.g. , C)
(B) Object-oriented language (e.g., C++)
(C) Assembly language
(D) The above can solve the same set of problems.
(E) The answer depends on compilers and operating systems.

__**B**__ To store a polynomial **f(x, y, z)** = $2x^3y^4 + 3x^2z^3 - 7xy^2z^2 + ...$ using an array.

(A) array size = 3 * terms
(B) array size = 4 * terms
(C) array size = $terms^2$
(D) array size = $terms^3$
(E) None of the above

__**C**__ int A[10][10][10] is an row-major array. What is the offset of A[3][3][3]?

(A) 3 * (1000 + 100) * sizeof(int)
(B) 3 * (1000 + 100 + 10 + 1) * sizeof(int)
(C) 3 * (100 + 10 + 1) * sizeof(int)
(D) 3 * (1000 + 100 + 10) * sizeof(int)
(E) None of the above

6. **(12pt)** Please convert **B * C + (A * B / C) – A / B + C** into a postfix expression**.** Only the boxes with thick borders will be graded (只有粗黑框格子計分). Please note **the priority listed in the table is not what we commonly use.**

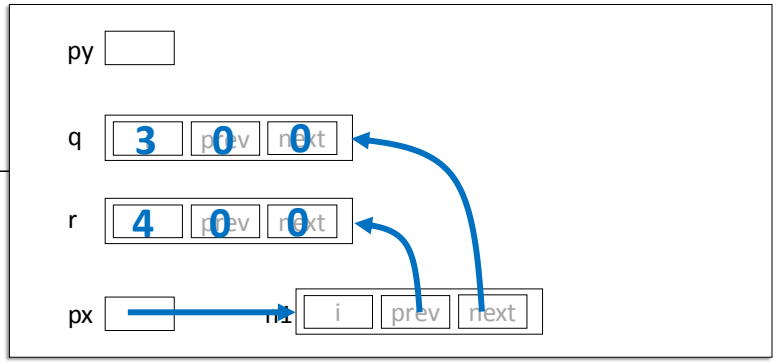
Priority	Operator
High	+
	-
Low	* /

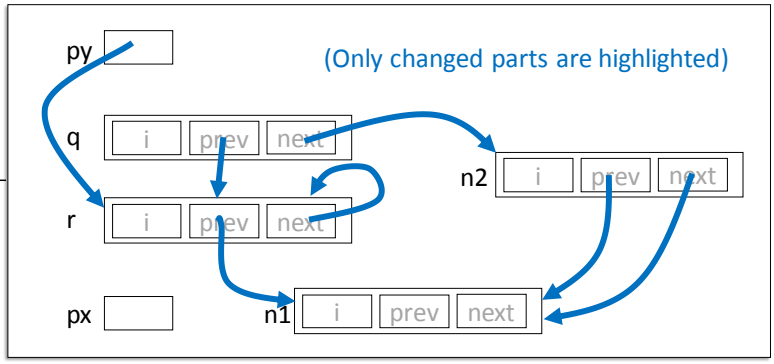
← bottom

Token	Stack						Output So Far															
B							B															
*****	*						B															
C	*						B	C														
+	*	+					B	C														
(	*	+	(				B	C														
A	*	+	(				B	C	A													
*****	*	+	(	*			B	C	A													
B	*	+	(				B	C	A	B												
/	*	+	(	/			B	C	A	B	*											
C	*	+	(				B	C	A	B	*	C										
)	*	+					B	C	A	B	*	C	/									
-	*	-					B	C	A	B	*	C	/	+								
A	*						B	C	A	B	*	C	/	+	A							
/	/						B	C	A	B	*	C	/	+	A	-	*					
B	/						B	C	A	B	*	C	/	+	A	-	*	B				
+	/	+					B	C	A	B	*	C	/	+	A	-	*	B				
C							B	C	A	B	*	C	/	+	A	-	*	B	C			
	Final output						B	C	A	B	*	C	/	+	A	-	*	B	C	+	/	

7. (10pt) Linked list operations

```
struct Node{
  int i;  Node* prev;  Node* next;
  Node(int i, Node* prev, Node* next):i(i), prev(prev), next(next) { }
};
Node * px, *py;
Node q(3, 0, 0), r(4, 0, 0);
px = &r;
px = new Node(10, px, &q);  // assume n1 is allocated
px->next->next = new Node(20, px, px); // assume n2 is allocated
q.prev = &r;
r.prev = px;
q.prev->next = q.prev;
py = px->next->prev->next;
```





8. (10pt) Asymptotic notations

$$P(n) = \boldsymbol{a_0} + \boldsymbol{a_1 n} + \boldsymbol{a_2 n^2} + \cdots + \boldsymbol{a_k n^k} \quad \Rightarrow \quad \log\big(P(n)\big) = \boldsymbol{O}(n)$$

(1) **(4pt)** Is the above statement **true or false**?

True

(2) **(6pt)** Please prove or disprove the statement **according to the definition of Big-O**.

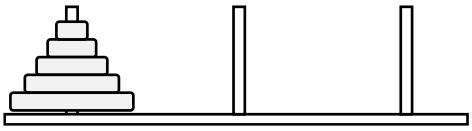
We shall prove:
∃ positive integers *C* and N_0 such that $log\big(P(n)\big) \le C \cdot n \quad \forall n \ge N_0$

It is evident that
$n \le n^k \qquad \forall n \ge 1 \text{ and } k \ge 1$
$\log(n) \le n \qquad \forall n \ge 1$

Let $C = (k+1)$ and $N_0 = \left(\sum_{i=0}^{k} |a_i|\right)$

$\Longrightarrow P(n) \le |a_0|n^k + |a_1|n^k + |a_2|n^k + \cdots + |a_k|n^k = N_0 n^k \qquad \forall n \ge 1$
$\Longrightarrow log\big(P(n)\big) \le (log(N_0) + klog(n)) \le C\, log(n) \le C \cdot n \quad \forall n \ge N_0$
Q.E.D.

9. **(11pt) Hanoi Tower**



(1) **(5pt)** Suppose the smallest disk weighs 1 gram, the 2nd smallest one weighs 2 grams, …, the Nth one weighs 2^{N-1} grams, how many grams are lifted in total to move the N disks to another stick?

weight(N) = 2^{N-1} + 2 weight(N-1)

= 2^{N-1} + 2^{N-1} + 4weigth(N-2)

= 2^{N-1} + 2^{N-1} + 2^{N-1} + 8 weigth(N-3)

= 2^{N-1} + 2^{N-1} + … + 2^{N-1} weigth(1) (total N terms)

= **$N*2^{N-1}$**

(2) **(6pt)** Please write a **recursive program** that simulates Hanoi Tower to calculate the above value.

```
int weight(int N)
{
   if(N==1) return 1;

   return (weight(N-1) + power(2, N-1) + weight(N-1));
}
```

10. **(9pt)**

(1) **(3pt)** Please describe two differences between **call-by-value** and **call-by-pointer**?

1. Call-by-pointer allows *side effect*, which means a *callee* can modify the variables residing in a *caller*. In contrast, call-by-value guarantees no such side effect.

2. Call-by-value copies arguments into the context of the *callee*. So, if arguments comprise large objects, the copy overhead is large. In comparison, call-by-pointer only copyies the addresses of objects and the overhead is low.

(2) **(3pt)** Please explain two purposes of **templates** in C++.

1. Avoid programmers from writing duplicate functions and classes code only handle algorithms (e.g., sorting) and data structures (e.g., linked lists) with different data types (e.g., int and float).

2. Allow algorithms (e.g., sorting) and data structures (e.g., linked lists) to handle user-defined data types (e.g., rectangle).

(3) **(3pt)** A multiplication operation is usually slower than an addition operation, but why can we still consider them the same when performing step count analysis?

Step count analysis focuses on understanding the trend that the execution time of an algorithm increases with the input size (problem size). The execution time of multiplication and addition operations are independent of input size, so we can consider them the same without affecting the analyzing results.