國立成功大學

112學年度碩士班招生考試試題

編　號：240

系　所：資訊管理研究所

科　目：計算機概論

日　期：0207

節　次：第 2 節

備　註：不可使用計算機

國立成功大學 112 學年度碩士班招生考試試題

系　　所：資訊管理研究所

考試科目：計算機概論

考試日期：0207，節次：2

※ 考生請注意：本試題不可使用計算機。　請於答案卷(卡)作答，於本試題紙上作答者，不予計分。

1. (15%)　True or False: Are each of the following True or False?

1.1. Dynamic programming provides an O(n) solution to the 0/1 knapsack problem, where n is the number of items to be considered.

1.2. K-means clustering is more computationally efficient than hierarchical clustering.

1.3. The depth of a decision tree is related to the number of independent decisions it records.

1.4. Hierarchical clustering is deterministic, but k-means clustering is not.

1.5. When used to find a root of a polynomial, Newton's method converges in O(log(n)) iterations, where n is the degree of the polynomial.

2. (15%) Multiple choice question: (choose only ONE answer for each question)

2.1 Which of the following is NOT a universal property of reader-writer locks?

(a) Readers can only look at a shared item; writers can also modify it.

(b) If a writer has access to the item, then no other thread also has access.

(c) Any number of readers can read the item at the same time.

(d) A writer waiting for an RW lock will get preference over subsequent read requests.

2.2 Starvation (in relation to threads) refers to:

(a) A thread waiting for a lock indefinitely.

(b) A semaphore that gets locked but the thread never unlocks it after use.

(c) A thread is spawned but never joins the main thread when finished.

(d) A process fails to spawn a new thread because it's hit the maximum number of threads allowed.

2.3 How does x86 assembly store the return value when a function is finished?

(a) The ret instruction stores it in a special retval register.

(b) By convention, it is always in %eax.

(c) It is stored on the stack just above the (%ebp) of the callee.

(d) It is stored on the stack just above all the arguments to the function.

2.4 In IEEE floating point, what would be an effect of allocating more bits to the exponent part by taking them from the fraction part?

(a) You could represent fewer numbers, but they could be much larger.

(b) You could represent the same numbers, but with more decimal places.

(c) You could represent both larger and smaller numbers, but with less precision.

(d) Some previously representable numbers would now round to infinity

2.5 Consider the following two blocks of code, found in separate files:

```
/* main.c */
int i=0;
int main()
{
  foo();
  return 0;
}
```

```
/* foo.c */
int i=1;
void foo()
{
  printf(''%d'', i);
}
```

What will happen when you attempt to compile, link, and run this code?

(a) It will fail to compile.

(b) It will fail to link.

(c) It will raise a segmentation fault.

(d) It will print "0".

(e) It will print "1".

(f) It will sometimes print "0" and sometimes print "1"

3. (20%) In this problem we consider properties of floating point operations. For each property state whether it is true or false. If false, give a counterexample as a (possibly negative) power of 2 within the range of precision for the variables. We assume that the variables on an x86_64 architecture are declared as follows

float x,y,z;
double d,e;

and initialized to some unknown value **different from NaN, +1, and−1**. We have given the first answer as an example.

$(x + y) + z == x + (y + z)$	**false**	$x = 1, y = 2^{127}, z = -2^{127}$
If $x > 0$ then $x / 2 > 0$		
$(x + y) * z == x * z + y * z$		
If $x >= y$ and $z <= 0$ then $x * z <= y * z$		
If $x > y$ then (double) $x >$ (double) y		
If $d > e$ then (float) $d >$ (float) e		
$x + 1 > x$		

※ 考生請注意：本試題不可使用計算機。 請於答案卷(卡)作答，於本試題紙上作答者，不予計分。

4. Short Answer (15%)

a) Suppose we have the following array:

8 15 3 1 14 23 6 10

We run a sorting algorithm on it, and the algorithm prints the updated array after each change. If at some point it prints the following, which algorithm is it? (3%)

1 3 8 15 6 10 14 23

b) Suppose we have a general tree and we want to print it layer by layer. That is, print the root node at first, follow by all nodes at depth 1, then all nodes at depth 2. What strategy should we use? (2%)

c) Suppose a CPU uses 8 bytes for its memory addresses. How many bytes of memory can it address? (2%)

d) Suppose you are asked to develop a text editor and you want to implement an "undo" feature. What data structure should you use in this case? (2%)

e) Write the 1-byte hex number A7 in binary, in unsigned decimal number, and in signed decimal number. (3%)

f) Suppose we have an array that is mostly sorted. Would insertion sort or selection sort be better, or would they have the same performance to obtain a totally sorted array? Why? (3%)

5. For the following algorithms, express the running time using big-O notation and briefly explain your answer. (15%, 5% each)

a) Algorithm A1(n)

```
i ← 1
while i < n
    for j = i to n do
        print(j)
    i ← i + 2
```

b) Algorithm A2(n)

```
i ← 1
while i < n
    print(i)
    i ← i + 5
while i > 1
    print(i)
    i ← i / 2
```

c) Algorithm A3(n)

```
i ← 1
while i < n
    for j = 1 to i do
        print(j)
    i ← i * 2
```

6. Write exactly one letter that best matches each item in the table below. No letter should be used more than once. (20%, 2% each)

Feature vector	Hierarchical clustering	Depth first search	Greedy algorithm	Data abstraction
(a)	(b)	(c)	(d)	(e)
Big-O notation	Polymorphism	Recursion	Hashing	Merge sort
(f)	(g)	(h)	(i)	(j)

A. Optimization	B. Undirected graph	C. Expected running time	D. Mutability
E. O(1)	F. Local optima	G. Induction	H. Upper bound
I. Specification	J. O(log n)	K. Inheritance	L. Unit testing
M. Normalization	N. Lower bound	O. Linkage criterion	P. O(n)
Q. Backtracking	R. Standard deviation	S. Divide and conquer	T. Approximation