Computer Science and Engineering 373

Data Structures and Algorithms

Steve Tanimoto, Instructor
Autumn Quarter, 2004

Midterm 1

Instructions This is a closed-book exam. Do not use any notes, books, calculators or computers. There are two parts. Part 1 contains 12 multiple choice questions worth 4 points each. Part 2 contains three questions worth a total of 32 points.

Part I. (48 points)

Put all your answers for this part on the multiple-choice answer sheet. Write your name on the answer sheet and code it into the bubble grid. Use only standard number 2 pencils for this part. For each question, choose the best answer. Mark only one answer per question. (A correct answer is worth 4, and an incorrect answer is worth 0).

If you feel that a question is flawed, inconsistent or overly ambiguous, give a clear explanation of your objection on the written-answer portion of the exam.

Good luck and do your best!

1. Consider the two sets $S_1 = \{a, b\}$ and $S_2 = \{0, 1, 2\}$. How many elements are there in the cartesian product $S_1 \times S_2$?

   [A] 1
   [B] 4
   [C] 5
   [D] 6
   [E] 8

2. Consider the two sets $S_1 = \{a, b\}$ and $S_2 = \{0, 1\}$. Which of the following elements is a member of the cartesian product $S_1 \times S_1 \times S_2$?

   [A] $\{(a, b, 0, 1)\}$
   [B] $(a, b, 0, 1)$
   [C] $((a, a), 1)$
   [D] $\{(a, b, 0, 1)\}$
   [E] $(a, b, 0)$

3. Consider the binary relation $R = \{(a, a), (a, b), (b, c)\}$ over the set $S = \{a, b, c, d\}$. Then R has which property?

[A] reflexive
[B] symmetric
[C] antisymmetric
[D] transitive
[E] total ordering

4. Suppose we are given $R_1 = \{(a, a), (a, b), (a, c), (d, d), (d, e), (e, d)\}$ and we are told that another relation R_2 is the same as R_1 except that it has one more ordered pair, and it is transitive. What must that ordered pair be?

   [A] (b, b)
   [B] (b, c)
   [C] (c, a)
   [D] (c, d)
   [E] (e, e)

5. Choose the symmetric relation:

   [A] $\{(a, b), (a, c), (c, a)\}$
   [B] $\{(a, a), (b, b), (c, c), (d, d), (a, b), (b, a)\}$
   [C] $\{(a, b), (b, c), (a, c)\}$
   [D] $\{(a, b), (c, d)\}$
   [E] $\{(a, b), (b, c), (c, d), (a, c)\}$

6. Which of the following relations is a function on the domain $\{a, b, c\}$ and some range?

   [A] $\{(a, b), (a, c), (c, a)\}$
   [B] $\{(a, a), (b, b), (c, c), (a, b), (b, a)\}$
   [C] $\{(a, b), (b, c)\}$
   [D] $\{(a, b), (b, c), (c, d)\}$
   [E] $\{(a, b), (b, c), (c, d), (a, c)\}$

7. Which description corresponds to the surjection property?

   [A] every domain element occurs as a left-hand side
   [B] every range element occurs as a right hand side
   [C] domain element has more than one range element
   [D] no range element has more than one domain element
   [E] none of these

8. The domain $D = \{a, b, c\}$ and the range $R = \{0, 1\}$. Which of these functions is a bijection? (That is, which of these is invertible?)

   [A] $\{(a, 0), (b, 0), (c, 0)\}$
   [B] $\{(a, 1), (b, 1), (c, 1)\}$
   [C] $\{(a, 0), (b, 0), (c, 1)\}$
   [D] $\{(a, 0), (b, 1), (c, 0)\}$
   [E] none

9. Suppose we represent the methods in a stack abstract data type using functional notation.

   Let the data elements in the stack be $\langle e_0, e_1, \cdots, e_{n-1}\rangle$, where e_0 is the top element in the stack. Which of the following best represents the PUSH operation?

[A] $f_{\text{PUSH}}(e, \langle e_0, e_1, \cdots, e_{n-1}\rangle) \mapsto \langle e_0, e_1, \cdots, e_{n-1}, e\rangle)$
[B] $f_{\textbf{PUSH}}(e, \langle e_0, e_1, \cdots, e_{n-1}\rangle) \mapsto \langle e, e_0, e_1, \cdots, e_{n-1}\rangle)$
[C] $f_{\text{PUSH}}(e, \langle e_0, e_1, \cdots, e_{n-1}\rangle) \mapsto \langle e_1, \cdots, e_{n-1}\rangle)$
[D] $f_{\text{PUSH}}(e, \langle e_0, e_1, \cdots, e_{n-1}\rangle) \mapsto \langle e_0, e_1, \cdots, e_{n-2}\rangle)$
[E] $f_{\text{PUSH}}(e, \langle e_0, e_1, \cdots, e_{n-1}\rangle) \mapsto \langle e, e_1, \cdots, e_n\rangle)$

10. Assume that we implement two stacks S_1 and S_2. For S_1 we use an array with enough space so that it doesn't overflow in our application, and for S_2 we use a linked list. Assuming that our implementations are written with equal diligence to time efficiency, what can we say about the running times $T_1(n)$ and $T_2(n)$ for the PUSH operation using S_1 and S_2, respectively? Let n be the number of elements currently in the stack.

    [A] $\Theta(n)$, $O(\log n)$
    [B] $\Theta(1)$, $\Omega(n)$
    [C] $\Omega(n)$, $\Omega(n)$
    [D] $O(n)$, $\Omega(n)$
    [E] $\Theta(1)$, $\Theta(1)$

11. Consider the following piece of Java code:

```
int k = 0;
for (int i = 0; i < 10; i++) {
  for (int j = 0; j < n; j++) {
    k++;
  }
}
```

    What expression correctly gives the final value of k as a function of n?

    [A] $k(n) = 10n$
    [B] $k(n) = (n^2 - n)/2$
    [C] $k(n) = 10^n$
    [D] $k(n) = n^2$
    [E] $k(n) = n\lceil\log_2\rceil n$

12. Suppose the running time of some algorithm is $T(n) = 23 + 17\log_3 n^2$. Recall that $f(n)$ is $\Theta(g(n))$ provided $f(n)$ is $O(g(n))$ and $g(n)$ is $O(f(n))$. Then we can say that $T(n)$ is

    [A] $\Theta(1)$
    [B] $\Theta(\log n)$
    [C] $\Theta(n)$
    [D] $\Theta(n \log n)$
    [E] $\Theta(2^n)$

Part II.

Do each of the following three problems.

1. (10 points) Let $f(n) = 1 + 3 + 5 + \cdots + 2n - 1$. Let $g(n) = n^2$. Prove by induction that for all integers $n \geq 1$,

$$f(n) = g(n)$$

Clearly mark your basis, induction hypothesis, and induction step. Hint: $f(n) = \sum_{k=1}^{n} 2k - 1$.

Basis:
$f(1) = \sum_{k=1}^{1} 2k - 1 = 2(1) - 1 = 2 - 1 = 1$
$g(1) = 1^2 = 1$
$1 = 1$
Inductive Hypothesis:
Assume that for some n, $f(n) = g(n)$. In other words $\sum_{k=1}^{n} 2k - 1 = n^2$
Inductive Step:
We assume the inductive hypothesis $f(n) = g(n)$ and from this we will show that $f(n+1) = g(n+1)$.
$1 + 3 + 5 + \cdots + 2n - 1 = n^2$
Now add the next term $2(n+1) - 1$ to both sides of the equation to get:
$1 + 3 + 5 + \cdots + 2(n+1) - 1 = n^2 + 2(n+1) - 1$
Note that the left hand side is now $f(n+1)$. The right hand side can be re-expressed by simplifying and then factoring.

$$\begin{aligned} 1 + 3 + 5 + \cdots + 2(n+1) - 1 &= n^2 + 2(n+1) - 1 \\ &= n^2 + 2n + 1 \\ &= (n+1)(n+1) \\ &= (n+1)^2 \\ &= g(n+1) \end{aligned}$$

So $f(n+1) = g(n+1)$ and we have proved the induction step. The basis together with the induction step prove that $f(n) = g(n)$ for all $n > 0$.

2. (10 points) Consider the binary relation R on $S = \{a, b, c\}$, where $R = \{(a,a), (a,b), (b,c)\}$ Is R a partial order over S? Why or why not? If it is not, identify what pairs must be added to make it so.

No, this is not a partial order. In order to be a partial order it needs to be reflexive, antisymmetric and transitive. This relation is not reflexive or transitive but it is antisymmetric. In order to make it reflexive we would need to add $\{(b,b), (c,c)\}$. In order to make it transitive we would need to add $\{(a,c)\}$.

3. (12 points) Suppose that a queue is to be represented using an array A with a maximum capacity of 16. The elements of A are denoted A[0], A[1], ..., A[15]. The head of the queue is given by an integer HEAD in the range 0 to 15. The number of elements is given by COUNT. An efficient way to implement the ENQUEUE and DEQUEUE operations allows each operation to be completed in $O(1)$ time. Describe this efficient implementation of the ENQUEUE operation. You may use English, Pseudocode, or Java to describe the method. You will be graded on getting the logic

of the method (not the syntax) correct. You may make the following assumptions: (a) that the queue is not at maximum capacity, and so there is room for at least one more element somewhere in the array; (b) that the data elements in the queue are general objects.

In order to manage a queue and keep *both* your enqueue and dequeue operations 0(1) **you need to implement a circular array.**

```
enqueue(o : object) : {
  index : integer;
  index := (HEAD + COUNT) mod 16;
  A[index] = o;
}
```