Exam Imperative Programming

Friday, December 6, 2019, 19:00 h.

- You can earn 90 points. You will get 10 points for free. So, you can obtain 100 points in total, and your exam grade is calculated by dividing your score by 10.
- This exam consists of 5 problems. The first two problems are multiple choice questions. The problems 3, 4, and 5 are made using a computer. All problems are assessed by the Themis judging system. For each of the problems 3, 4 and 5, there are 10 test cases. Each test case is worth 10% of the points.
- Note that manual checking is performed after the exam. For example, if a recursive solution is requested then a correct iterative solution will be rejected after manual checking, even though Themis accepted it. Also, precomputed answers will be rejected.
- This is an open book exam! You are allowed to use the pdf of the reader (which is available in Themis), pdfs of the lecture slides (also available in Themis), the prescribed ANSI C book (hard copy) and a dictionary. Any other documents are not allowed. You are allowed to use previous submissions that you made to Themis.

Problem 1: Assignments (20 points)
For each of the following annotations determine which choice fits on the empty line (.....). The variables `x`, and `y` are of type `int`. Note that `A` and `B` (uppercase letters!) are specification constants (so not program variables).

1.1
```
/* x == A, x + y == B */
.....
/* x - y == A, x == B */
```

(a)
```
x = x - y;
```
(b)
```
x = x + y; y = x - A;
```
(c)
```
x = x + y;
```

1.2
```
/* x - y == A, x == B */
.....
/* x == A, x + y == B */
```

(a)
```
y = 0;
```
(b)
```
x = x - y;
```
(c)
```
x = x - y; y = x + y;
```

1.3
```
/*  x == A + B, y == A - B */
.....
/* x == 4*A, y == 5*A - 3*B */
```

(a)
```
x = 2*(x + y); y = x + y;
```
(b)
```
y = y-2*x; x = 6*x+2*y; y = x+y;
```
(c)
```
y = 2*y - x; x = y+3*x; y = x+y;
```

1.4
```
/* x == A, y == B */
x = 2*x + y; y = 2*x + y;
.....
```

(a)
```
/* x == 2*A + B, y == 2*A + B */
```
(b)
```
/* x == 2*A + B, y == 4*A + 3*B */
```
(c)
```
/* 2*x == A - B, y == 2*B + A */
```

1.5
```
/* x == A, y == B */
x = 2*x + 2*y; y = x - 2*y; x = x - y;
.....
```

(a)
```
/* x == 2*B, y == 2*A  */
```
(b)
```
/* x == B, y == 2*A */
```
(c)
```
/* 4*x == 3*A - 8*B, 4*y == A - 4*B */
```

1.6
```
/* x + y == A, 2*x + y == B */
x = x + 1; y = y - 1;
.....
```

(a)
```
/* x + y == A, 2*x + y == B */
```
(b)
```
/* x + y == A, 2*x + y - 1 == B */
```
(c)
```
/* x + y == A, 2*x + y + 1 == B */
```

Problem 2: Time complexity (20 points)
In this problem the specification constant `N` is a positive integer (i.e. `N>0`). Determine for each of the following program fragments the *sharpest upper limit* for the number of calculation steps that the fragment performs in terms of `N`. For a fragment that needs `N` steps, the correct answer is therefore $O(N)$ and not $O(N^2)$ as $O(N)$ is the sharpest upper limit.

1. ```
   int s = 0;
   for (int i=0; i < N; i++) {
     int k = 2 + i%5;
     for (int j=1; j < N; j*=k) {
       s += i + j;
     }
   }
   ```
   (a) $O(\log N)$ (b) $O(\sqrt{N})$ (c) $O(N)$ (d) $O(N \log N)$ (e) $O(N^2)$

2. ```
   int i=0, j=0, s=0;
   while (i*j < 2*N) {
     i = i + 1;
     j = j + 2;
     s = s + i + j;
   }
   ```
   (a) $O(\log N)$ (b) $O(\sqrt{N})$ (c) $O(N)$ (d) $O(N \log N)$ (e) $O(N^2)$

3. ```
   int s = 0, i = 0;
   while (s < N*N) {
     s = s + i;
     i++;
   }
   ```
   (a) $O(\log N)$ (b) $O(\sqrt{N})$ (c) $O(N)$ (d) $O(N \log N)$ (e) $O(N^2)$

4. ```
   int i = 1, s = 0;
   while (s < N) {
     s = s + i;
     i += i + i%2;
   }
   ```
   (a) $O(\log N)$ (b) $O(\sqrt{N})$ (c) $O(N)$ (d) $O(N \log N)$ (e) $O(N^2)$

5. ```
   int i=1, s=0;
   while (i < N*N) {
     i = 2*i;
   }
   while (i > 0) {
     s += i;
     i--;
   }
   ```
   (a) $O(\log N)$ (b) $O(\sqrt{N})$ (c) $O(N)$ (d) $O(N \log N)$ (e) $O(N^2)$

6. ```
   int i=1, s=0;
   while (i*i < N) {
     i = 2*i;
   }
   while (i>0) {
     for (int j=0; j < i; j++) {
       s += i + j;
     }
     i--;
   }
   ```
   (a) $O(\log N)$ (b) $O(\sqrt{N})$ (c) $O(N)$ (d) $O(N \log N)$ (e) $O(N^2)$

Problem 3: Number of Pair Sums (15 points)
The input of this problem consists of two positive integers n and k, followed by a series of n non-negative integers without duplicates (see examples). The output should be the number of pairs (a, b), where a and b are drawn from the series, such that $a + b = k$ and $a \neq b$. Note that the pair (a, b) is considered to be the same pair as (b, a).

Example 1:
input:
```
5 42:10,15,20,22,27
```
output:
```
2
```

Example 2:
input:
```
8 10:1,2,4,3,7,8,6,0
```
output:
```
3
```

Example 3:
input:
```
6 42:11,13,17,19,23,29
```
output:
```
2
```

Problem 4: Horner representation (15 points)
The Horner representation of a polynomial expression is named after William George Horner. The Horner representation of the polynomial expression $a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots + a_nx^n$ is given by

$$(((\cdots(a_nx + a_{n-1})x + \cdots a_3)x + a_2)x + a_1)x + a_0$$

As a concrete example, the Horner representation of the polynomial expression $8 + x - 7x^2 + 4x^3 + 6x^5$ is

$$((((6x + 0)x + 4)x - 7)x + 1)x + 8$$

Note that in the polynomial expression, the term that corresponds with x^4 is missing. Hence $a_4 = 0$. Moreover, note that the coefficient that corresponds with x^2 is negative ($a_2 = -7$).

The input of this problem consists of a positive integer n followed by a polynomial expression which is a sum of n terms of the form `a*x^b` where `a` and `b` are integers and $0 \leq$ `b` ≤ 100. The terms may be in any order (compare the examples 1 and 2). There may be several occurrences of terms with the same degree (see example 3). Your program should output the Horner representation of the polynomial expression that is given on the input.

Example 1:
input:
```
5 8*x^0+1*x^1-7*x^2+4*x^3+6*x^5
```
output:
```
((((6*x+0)*x+4)*x-7)*x+1)*x+8
```

Example 2:
input:
```
5 6*x^5+8*x^0-7*x^2+4*x^3+1*x^1
```
output:
```
((((6*x+0)*x+4)*x-7)*x+1)*x+8
```

Example 3:
input:
```
8 1*x^3+2*x^3+3*x^3+1*x^0+20*x^0+36*x^3+21*x^0+42*x^2
```
output:
```
((42*x+42)*x+0)*x+42
```

Problem 5: Sum of Subsets (20 points)
Consider the series $[1, 2, 3]$. Ignoring the order of the numbers (i.e. $[1, 2, 3]$ is the same as for example $[3, 1, 2]$), the series has seven non-empty subseries: $[1]$, $[2]$, $[3]$, $[1, 2]$, $[1, 3]$, $[2, 3]$, $[1, 2, 3]$.
If we sum the values in each of these subseries, and sum these sums, we find the *sum of subsets* of the series $[1, 2, 3]$:

$$1 + 2 + 3 + (1 + 2) + (1 + 3) + (2 + 3) + (1 + 2 + 3) = 1 + 2 + 3 + 3 + 4 + 5 + 6 = 24.$$

Write a program that reads from the input an integer n (where $1 < n \leq 20$) and outputs the sum of subsets of the series containing the integers 1 upto n, i.e. the series $[1, 2, 3, .., n]$.

Example 1:
input:
```
3
```
output:
```
24
```

Example 2:
input:
```
5
```
output:
```
240
```

Example 3:
input:
```
10
```
output:
```
28160
```