COMP9024 18s2

Week 02 Problem Set
Elementary Data and Control Structures in C

Data Structures and Algorithms

1. (Arithmetic)

   There is a 5-digit number that satisfies 4·*abcde* = *edcba*, that is, when multiplied by 4 yields the same number read backwards. Write a C-program to find this number.

2. (Arrays)

   a. Write a C-function that returns the *inner product* of two *n*-dimensional vectors **a** and **b**, encoded as 1-dimensional arrays of *n* floating point numbers.

      Use the function prototype **float innerProduct(float a[], float b[], int n)**.

      Hint: The inner product of two vectors is calculated as $\sum_{i=1..n} \mathbf{a}_i \cdot \mathbf{b}_i$

   b. Write a C-function to compute **C** as the *matrix product* of matrices **A** and **B**.

      Use the function prototype **void matrixProduct(float a[M][N], float b[N][P], float c[M][P])**. You can assume that *M, N, P* are given as symbolic constants, e.g.

```
#define M 3
#define N 4
#define P 4
```

      Hint: The product of an *m*×*n* matrix **A** and an *n*×*p* matrix **B** is the *m*×*p* matrix **C** such that $\mathbf{C}_{ij} = \sum_{k=1..n} \mathbf{A}_{ik} \cdot \mathbf{B}_{kj}$ for all *i*∈{*1..m*} and *j*∈{*1..p*}.

3. (Characters)

   Write a C-program that outputs, in alphabetical order, all strings that use each of the characters 'c', 'a', 't', 'd', 'o', 'g' exactly once.

   How many strings does your program generate?

4. (Elementary control structures)

   a. Write a C-function that takes a positive integer *n* as argument and outputs a series of numbers according to the following process, until 1 is reached:

      - if *n* is even, then *n* ← *n*/2
      - if *n* is odd, then *n* ← 3**n*+1

   b. The Fibonacci numbers are defined as follows:

      - Fib(1) = 1
      - Fib(2) = 1
      - Fib(*n*) = Fib(*n*-1)+Fib(*n*-2) for *n*≥3

      Write a C program `fibonacci.c` that applies the process described in Part a. to the first 10 Fibonacci numbers.

      The output of the program should begin with

```
Fib[1] = 1
1
Fib[2] = 1
1
Fib[3] = 2
2
1
Fib[4] = 3
3
10
5
16
8
4
2
1
```

We have created a script that can automatically test your program. To run this test you can execute the `dryrun` *program that corresponds to the problem set and week, i.e. prob02 for this week. It expects to find a program named* `fibonacci.c` *in the current directory. You can use dryrun as follows:*

```
prompt$ ~cs9024/bin/dryrun prob02
```

Note: Please ensure that your output follows exactly the format shown above.

5. (Elementary data structures)

   Define a data structure to store all information of a single ride with the Opal card. Here are three sample records:

Transaction number	Date/time	Mode	Details	Journey number	Fare Applied	Fare	Discount	Amount
2013	Mon 30/07/2018 10:16	B	Flinders St af Oxford St to Anzac Pde D opp UNSW	1		$1.46	$0.00	-$1.46
2011	Mon 30/07/2018 10:05	B	Victoria St at Liverpool to Oxford St op Palmer St	1		$2.20	$0.00	-$2.20
2009	Sun 29/07/2018 17:35	T	Bondi Junction to Kings Cross		Day Cap	$3.54	$3.54	$0.00

   You may assume that individual stops (such as "Anzac Pde D opp UNSW") require no more than 31 characters.

   Determine the memory requirements of your data structure, assuming that each integer and floating point number takes 4 bytes.

   If you want to store millions of records, how would you improve your data structure?

6. **Challenge Exercise**

   Write a C-function that takes 3 integers as arguments and returns the largest of them. The following restrictions apply:

   - You are not permitted to use **if** statements.
   - You are not permitted to use loops (e.g. **while**).
   - You are not permitted to call any function.
   - You are only permitted to use data and control structures introduced in Week 2's lecture.