CENG 2010 Programming Language Concepts, Spring 2021
FINAL EXAM
DATE: 14.06.2021
10:30-12.30
50% OF THE OVERALL GRADE

Write your answers to a Word doc and convert it to a pdf before uploading. HAND_WRITTEN format is also accepted in case you feel more comfortable!!!

Q1- 10 pts

True/False Questions. Write your asnwers as: Q1-1: correct answer, Q1-2: correct answer, etc….

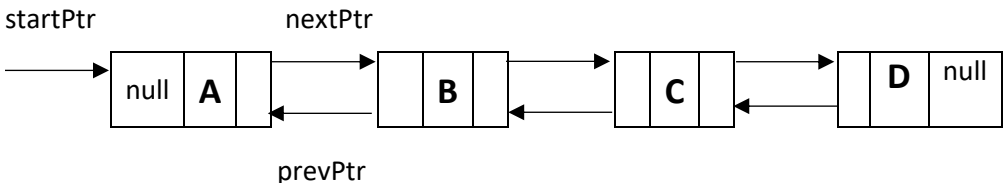
1. T / F-Disjoint union assuming that variants are chosen from type S or type T = (#S)Λ (#T)
2. T / F -Cardinality of mapping S→T is: (#*S*)x(#T) →$(\#T)^{\#S}$
3. T / F - Smalltalk and Lisp are dynamically typed PLs whereas Prolog and Perl are statically typed.
4. T / F - If you can print a piece of code, or assign it to a variable, it's an expression. If you can't, it's a statement!
5. T / F - memcpy() copies specific number of bytes from source to destination in RAM, where as strcpy() copies a constant / string into another string.
6. T / F -The semantics of a PL describe the sequence of symbols that make up valid programs
7. T / F -Enumeration variables are typically implemented using a subrange of the integers.
8. T / F -Ambiguity is a desirable syntactic attribute.
9. T / F -The execution of programs developed in a compiled language is typically more efficient than the interpreted ones.
10. T / F - One common approach to storage for a variant record type is to allocate adequate storage for the larger variant for all records of that type.

Q2-20 pts

Is it true that if a programming language has a while statement, it does not need an if statement? Explain your answer in detail.

Q3 – 30 pts – Bitwise XOR Linked List (XOR LL)

In this question, you are asked to write <u>two functions: insertNode and forwardTraverse</u>, for an XOR linked list, a.k.a, memory efficient doubly linked list (DLL) in C PL. Ordinary DLLs requires space for two addresses (one pointing to previous node and one to next node). However, XOR LLs use bitwise XOR operation to save space for one address. Consider the below ordinary DLL.



If you consider node B, prevPtr keeps the address of A, nextPtr keeps the address of C. Node A'a prevPtr and node D's nextPtr keep NULL.
Now, let's name the address variable as **npxor**.
For node A, **npxor** = 0 XOR (address of B)
For node B, **npxor** = (address of A) XOR (address of C)
and so on….

Now remember bitwise XOR table:

bit a	bit b	a XOR b
0	0	0
0	1	1
1	0	1
1	1	0

Forward Traversal of XOR LL: While traversing the list we need to remember the address of the previously accessed node in order to calculate the next node's address.

Remember this simple logic, e.g. for node C:

(address of B) XOR (npxor of C) = (address of D).

The reason is:

(address of B) XOR (address of D) = (npxor of C)

➔

(npxor of C) XOR (address of B) = (address of B) XOR (address of D) XOR (address of B)

= (address of D) XOR 0 = (address of D)

Node Insertion into XOR LL: Insert the node in the beginning. As you remember from the course, the head pointer should be manipulatable.

Given the related helper functions and node struct, fill the two empty functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <inttypes.h>

struct Node
{
    int data;
    struct Node* npx;
};

struct Node* XOR (struct Node *a, struct Node *b)
{
    return (struct Node*) ((uintptr_t) (a) ^ (uintptr_t) (b));
}

void insertNode(struct Node **headPtr, int data)
{
    // Allocate memory for new node
    /* Since new node is being inserted at the begining, npx of new node will always
       be XOR of current head and NULL */
    /* If linked list is not empty, then npx of current head node will be XOR
       of new node and node next to current head */
    // Change head

    //FILL THIS FUNCTION




}

void forwardTraverse (struct Node *headPtr)
{
    //FILL THIS FUNCTION




}
```

Q4 – 40 pts

Find two programming languages that are new to you, and answer the following questions for each:

- When was the language first introduced, and who developed it?
- Why was the language created? What novel about the language?
- Show and describe a fragment of code (in one of these languages) that you think is particularly interesting, and describe what it does and why it is interesting.
- Is the language still used today for anything?

**You should exclude the following languages from your search: C, C++, C#, Java, Javascript, FORTRAN, LISP, Pascal, Perl, Python, Racket.