I2P(II)2020 Yang Written Exam Practice (part2)

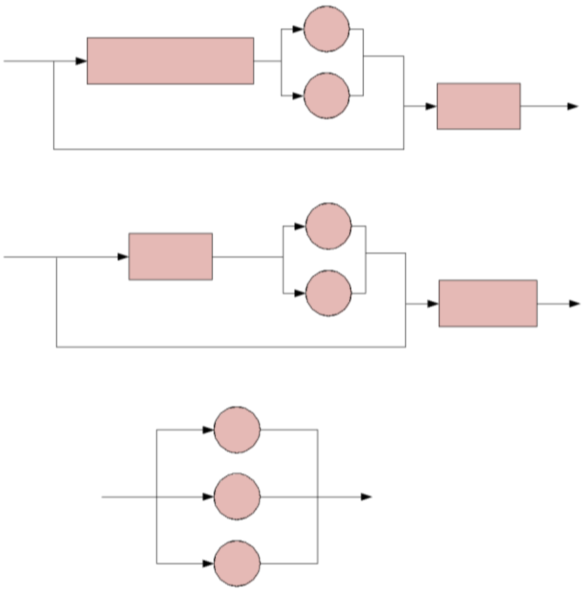
Mini Project 1&2

1. Draw the parse tree / syntax tree of the 2 inputs:
   (i) a = x + y * z
   (ii) a = x + y + z

2. Algebraic expressions manipulating variables x, y and z, such as “x-y*z+x/y”, can be described by the following grammar recursively:

Expression	:= Term | Expression ADDSUB Term
Term	:= Factor | Term MULDIV Factor
Factor	:= x | y | z

   where “|” means “or”, or equivalently by the following syntax diagrams



   For the string **x + y * z**, draw the parse tree based on the above syntax diagrams.

3. How can we reduce the clock cycles of the generated assembly code? Please provide two potential methods and briefly describe them.

4. Can we pre-calculate the results (values of x, y, z) directly for all inputs? Assuming we want the final value of x, y, z to be at r0, r1, r2.
   For example:
   x = 1

```
y = 2
z = x + y + 3
x = z - y
```

can be easily simplified to (x,y,z) = (4,2,6), that is

```
MOV r0 4
MOV r1 2
MOV r2 6
EXIT 0
```

If we can pre-calculate all the required results and reduce them into only 4 instructions, why bother generating the parser/syntax tree?

5. What is the final value of the registers (r0, r1) of the following instructions?

```
MOV r1 3
MOV r0 r1
EXIT 0
```

6. Please write out any assembly code that can represent following operation (assuming x, y, z values are at r0, r1, r2 initially and we want the final values also be at r0, r1, r2):

```
a = x + y * z
x = a
```

7. Can we use for-loop to achieve the same compile result as using recursion?

8. Briefly describe the most difficult problem you encountered in mini-project-1.

9. Briefly describe the most difficult problem you encountered in mini-project-2.

Trace Code - (C Style) Linked List

A doubly linked list is a linked data structure that consists of a set of sequentially linked nodes. Typically, a node can be implemented as follows:

```
// definition of Node structure
typedef struct Node {
    int value;
    struct Node *prv, *nxt;
} Node;
// memory allocation of Node structure
Node* new_node(int val) {
    Node *ptr = (Node*)malloc(sizeof(Node));
    ptr->value = val;
    ptr->prv = ptr->nxt = NULL;
    return ptr;
}
// declaration of pointer to head and rear
Node *head = NULL, *rear = NULL;
```

Let’s say we want to build a doubly linked list with an array of *n* elements. Doubly linked list can be built with several ways, some of them uses dummy nodes (nodes that do not actually store value). Here are three possible implementations:

```
#define DUMMY_VALUE 87878787
// implementation 1:
void impl_1(int *arr, int n) {
    head = new_node(arr[0]), rear = head;
    for(int i = 1; i < n; i++) {
        Node *ptr = new_node(arr[i]);
        rear->nxt = ptr, ptr->prv = rear;
        rear = ptr;
    }
}
// implementation 2:
void impl_2(int *arr, int n) {
    head = new_node(DUMMY_VALUE), rear = head;
    head->nxt = rear, rear->prv = head;
    for(int i = 0; i < n; i++) {
        Node *ptr = new_node(arr[i]);
        ptr->nxt = rear, ptr->prv = rear->prv;
        rear->prv->nxt = ptr;
        rear->prv = ptr;
    }
}
// implementation 3:
```

```
void impl_3(int *arr, int n) {
    head = new_node(DUMMY_VALUE), rear = new_node(DUMMY_VALUE);
    head->nxt = rear, rear->prv = head;
    for(int i = 0; i < n; i++) {
        Node *ptr = new_node(arr[i]);
        ptr->nxt = rear, ptr->prv = rear->prv;
        rear->prv->nxt = ptr;
        rear->prv = ptr;
    }
}
```

Q1

We want to build doubly linked lists with **5** elements {1,2,3,4,5} using the three implementations shown above.

Please draw how the doubly linked lists look like using the three implementations. Use circle to denote normal node and write its value in the circle; use triangle to denote dummy node; use arrow to denote how the nodes connect to each other; use "NULL" to denote NULL. Also please mark the nodes that represent head and rear respectively.

```
int main() {
    int arr[5] = {1, 2, 3, 4, 5};
    // call either one of the implementations
    impl_1(arr, 5);
    impl_2(arr, 5);
    impl_3(arr, 5);
}
```

Q2

A function is designed to print the contents of the doubly linked list (contents does not include the value of dummy nodes) :

```
void print_list(Node *begin, Node *end) {
    for(Node *ptr = begin; ptr != end; ptr = ptr->nxt)
        printf("%d ", ptr->value);
    printf("\n");
}
```

What values should be passed on to *print_list* function so that the contents of the doubly linked list can be properly printed?

i.e. How should we fill in the blanks (1a, 1b, 2a, 2b, 3a, 3b) ?

```
Node *head = NULL, *rear = NULL;
// ... declaration of Node structure
// ... implementation of impl_1, impl_2, impl_3
int main() {
    int arr[5] = {1, 2, 3, 4, 5};
    // use either one of the implementations
    impl_1(arr, 5);
    print_list(__1a__, __1b__);
    impl_2(arr, 5);
    print_list(__2a__, __2b__);
    impl_3(arr, 5);
    print_list(__3a__, __3b__);
}
```

Trace Code – OOP

```
#include <string>
#include <iostream>
using namespace std;

class Human{
public:
    Human(string name, int age): Name(name), Age(age){
        cout << "Human " << name << " created" << endl;
    }
    virtual void SelfIntroduce(){
        cout << "Hello! My name is " << this->Name;
        cout << ", " << this->Age << " years old." << endl;
    }
protected:
    string Name;
    int Age;
};

class Engineer : public Human{
public:
    Engineer(string name, int age): Human(name, age){}
    virtual void WriteCode(){
        cout << "Coding..." << endl;
    }
};

class Poorgramer : public Engineer{
public:
    Poorgramer(string name, int age): Engineer(name, age){}
    virtual void SelfIntroduce(){
        cout << "Hello! My name is " << this->Name;
        cout << ", " << this->Age << " years old." << endl;
        cout << "As a poorgramer, I have to write code day and night." << endl;
    }
};

int main(){
    Poorgramer Mike("Mike", 18);
    Mike.SelfIntroduce();
    Mike.WriteCode();
    return 0;
}
```

Q1

What is the output of the program above?

(A)

```
Human Mike created
Hello! My name is Mike, 18 years old.
As a poorgramer, I have to write code day and night.
```

(B)

```
Human Mike created
Hello! My name is Mike, 18 years old.
As a poorgramer, I have to write code day and night.
Coding...
```

(C)

```
Hello! My name is Mike, 18 years old.
As a poorgramer, I have to write code day and night.
Coding...
```

(D)

```
Human Mike created
Hello! My name is Mike, 18 years old.
Coding...
```

(E)

This program can’t be compiled.

```
#include <string>
#include <iostream>
using namespace std;

class Animal{
public:
    Animal(string name) : Name(name){}
    virtual void MakeSound() = 0 ;
private:
    string Name;
};
class Dog : public Animal{
    Dog(string name): Animal(name) {}
    void MakeSound(){
        cout << "bow-wow" << endl;
    }
};
```

```
class Cat : public Animal{
    Cat(string name): Animal(name) {}
    void MakeSound(){
        cout << "meow" << endl;
    }
};

int main(){
    Cat kitty( "kitty" );
    Dog doggy( "doggy" );
    kitty.MakeSound();
    doggy.MakeSound();
    return 0;
}
```

Q2

Check the code above, if you think it works well, write down the output of it. Otherwise, figure out the reason why it doesn’t work.