2021 년 2 학기 프로그래밍언어론 중간고사(1 분반)

학번: 201810993 이름: [illegible] 성적 공개용 ID: [illegible]

성적공개용 ID: 성적을 공개할 때, 본인 성적을 확인할 수 있도록 아이디를 한 개 정합니다. 전체 성적을 공개할 때, 개인 정보 보호를 위해 학번이나 이름대신, 이 아이디를 이용해서 공개합니다. 아이디는 영문자, 숫자, 한글만 조합해서 만듭니다(특수문자나 기호 사용 불가)

제출: e-campus 에 "중간고사 제출"에 pdf 로 변환해서 제출. 단 시험 종료 10 분 전~종료까지 e-campus 에 장애가 있는 경우 e-mail 로 제출 가능(ycho@smu.ac.kr)

답안 작성: 시험지 출력 못하면 빈 종이에 학번, 이름, ID, 답을 작성해서 스캔 또는 사진 찍고 pdf 로 변환 후 제출. 답안은 한글로 작성.

1. 동적 스코프(dynamic scope)를 사용하는 언어를 이용해서 다음 코드를 작성하고 실행한다고 가정한다. 얕은 바인딩(shallow binding)과 깊은 바인딩(deep binding)을 사용하는 경우, 화면에 출력되는 내용을 각각 보이고, 왜 그런 결과가 나타나는 지 설명한다. 참고로 write_integer() 함수는 인자로 전달된 정수값을 화면에 출력한다. (출력 결과 20 점, 설명 10 점).

```
int x; // 전역 변수
void set_x(int n) { x = n }
void print_x() {
    write_integer(x);
}
void second(func f) {
    void first() {
        print_x();
        set_x(7);
    }

    int x = 4;
    print_x();
    first();
    f();
}
void main() {
    set_x(5);
    print_x();
    second(print_x);
    print_x();
}
```

shallow binding	deep binding
5	5
4	4
4	4
7	5
5	5

해당 결과가 나오는 이유는 shallow binding은 호출되기 직전의 referencing environment를 사용하기에 첫번째에서는 전역 변수 x 인 5의 값을, 이후에는 local 의 x의 값인 4를 출력하고 x가 7로 변경된 후 출력후 다시 전역으로 오게 된다.

deep binding은 호출직전이 아닌 정의될 때 환경으로 하므로 5, 4, 4, 5, 5 의 결과가 나타난다.

2. closest nested scope rule 에 대해서 설명하고, 간단한 코드로 예를 보인다. 단 nested procedure 가 지원되는 경우와 지원되지 않는 경우에 대해 두 가지 예를 보인다. (30 점)

Closest nested scope rule은 가장 안쪽 즉 자신과 가까운 곳부터 scope를 확인하고 없으면 다음 scope를 적용하는 rule입니다.

```
전역 int n;

int main(void)
{
    n=1;
    void f()
    {
        int n=3;
    }
}
```

3. 복합 대입 연산자(combination assignment operator)를 사용하는 것이 일반적인 산술 연산자(arithmetic operator)와 대입 연산자(assignment operator)를 사용하는 것보다 나은 점(장점)에 대해서 설명하시오(25 점)

만약 일반적인 산술연산자와 대입연산자를 사용해서 ex) X.Y[7].sinx = X.Y[7].sinx+1 을 푼다면, X라는 배열의 Y라는 구조체의 배열 7번째 요소의 sinx라는 값을 계산하고 그값에 다시 이같은 작업을 반복해서 그값에 1을 더해 돌려주는 많은 계산을 하지만 복합 대입 연산자를 사용하면 해당 계산을 2회를 할 필요가 없다.

4. Java 나 C#에서는 definite assignment 을 보수적으로(conservatively) 지원한다. Definite assignment 이 무엇인지 설명하고, definite assignment 가 보수적으로 지원된다고 했을 때, 논리적으로 오류를 범할 수 있는 본인만의 코드(책이나 강의노트에 있는 코드를 사용하지 않고 본인만의 새로운 코드)를 작성하고, 왜 오류가 될 수 있는지 설명한다(20 점)

Definite assignment는 표현식에 사용되는 모든 변수값을 미리 assign 해야 하는 것을 말한다. 즉 초기화 하지 않은 변수들에 대한 처리 방법이다.

```
public void static main( )
{
    public class myInt()
    {
        Int i;
        double j;
        myInt()
        {
            this.i=1;
        }
        void set()
        {
            If (i>-3)
            {
                j= 1.0;
            }
        }
        void print()
        {
            System.out.println(j);
        }
    }
    myInt n1 =new A1();
    n1.set()
    n1.print()
}
```

해당 경우 오류가 발생할 것으로 예상된다. 이유는 set 함수에서 double j에 대한 값이 없어 보이지 우리는 알고 있지만 i가 1이므로, 그러나 compiler는 몰라 오류를 발생할 것이다.