1. Int \* <- char \* 가능

float \* = int \* | ABA // arithmetic <- integral

int \* = float \* | A // integral <- arithmetic

sem\_expression()

N\_EXP\_ASSIGN:

-> semantic\_error 주석처리함 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

isAllowableAssignmentConversion()

-> isCompatiblePointerType(t1, t2) 에서 false면 팅김 (t, f, f)

**-> isPointerType(t1) && isPointerType(t2)이면 warning 뜨게해서 =해주기.**

**유 ~~ 효~~**

isCompatiblePointerType()

-> isPointerType(t1) : 1, isPointerType(t2) : 1

isCompatibleType()

-> isPointerType(t1) : 0, isPointerType(t2) : 0

isPointerType()

printf("isPointerType(t1) : %d, isPointerType(t2) : %d\n", isPointerType(t1),isPointerType(t2));

// 두 포인터 타입이 같은 경우. i.e. int \* <- int \*나 float \* <- float \*

else if (isPointerType(t1) &&

(isConstantZeroExp(node) || isCompatiblePointerType(t1, t2)))

return (TRUE);

// 두 포인터 타입이 다른 경우. i.e. int \* <- float \*나 char \* <- int \*

else if (isPointerType(t1) && isPointerType(t2))

semantic\_warning(11, node->line);

4\_1\_cast\_scalar->integral\_implicitness.c

// i.e. integral\_type <- pointer\_type

else if (isIntegralType(t1) && isPointerType(t2))

semantic\_warning(11, node->line);

4\_3\_cast\_integral->pointer\_implicitness.c

// pointer\_type <- integral\_type

else if (isPointerType(t1) && isIntegralType(t2))

semantic\_warning(11, node->line);

1. 값의 초기화. 왜, int 초기화는 잘 되는 것인가? -> 그냥 그렇다. 교수님이 상관없다신다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Sem\_declaration()

-> if (id->init) // initializer

sem\_initializer(id->init, id->level); 를 추가함

void \*sem\_initializer(A\_NODE \*node) { A\_TYPE \*t; switch (node->name) { case N\_INIT\_LIST: sem\_initializer(node->llink); sem\_initializer(node->rlink); break; case N\_INIT\_LIST\_ONE: sem\_expression(node->clink); break; case N\_INIT\_LIST\_NIL: break; } }

* Sem\_ininitalizer() 함수를 추가하면 **float 초기화도 가능하다!**
* 물론, 초기화 구현하지 말라해서 하면 안된다.

3. N\_EXP\_CAST : 에 다음과 같이 추가했다.

3\_1\_cast\_scalar->integral\_explicitness.c

// i.e. int\_type <- enum\_type이 가능하도록 함.

else if (isAnyIntegerType(t1) && (isScalarType(t2)))

return (TRUE);

// i.e. enum\_type <- scalar\_type이 가능하도록 함.

else if (isIntegralType(t1) && (isScalarType(t2)))

return (TRUE);

3\_3\_cast\_integral->pointer\_explicitness.c

// i.e. pointer\_type <- enum\_type이 가능하도록 함.

else if (isPointerType(t1) && (isScalarType(t2)))

return (TRUE);