연습문제

9-1. 어떤 상황에서는 typedef를 #define으로 대치할 수 있다. 다음 예제를 보자.

- typedef float DOLLARS;
- int main(void)
- {
- DOLLARS amount = 100.00, interest = 0.07 * amount;
- printf("DOLLARS = %.2f₩n",amount + interest);
- }

이 프로그램을 실행해 보아라. 그리고, typedef를 다음으로 대치하여라.

#define DOLLARS float

수정된 프로그램을 다시 컴파일하고 수행해 보아라. 이전과 동일한 결과를 내는가?

Sol)

typedef float DOLLARS 의 경우

[romance@161s ch9_code]$ ./a.out
DOLLARS = 107.00
[romance@161s ch9_code]$

typedef DOLLARS float 의 경우

[romance@161s ch9_code]$ ./a.out
DOLLARS = 107.00
[romance@161s ch9_code]$

9-2. 어떤 상황에서는 typedef를 #define으로 대치할 수 없다. 다음 프로그램을 보자.

- typedef float DOLLARS;
- int main(void)
- {
- DOLLARS amount = 100.00,

```
-                   interest = 0.07 * amount;
- {
-     float DOLLARS;
-      DOLLARS = amount +  interest ;
-      printf("DOLLARS = %.2f₩n",DOLLARS);
- }
- return 0;
- }
```

이 프로그램을 수행해 보아라. 그리고 typedef를 다음으로 대치해 보아라.

#define DOLLARS float

이제 프로그램은 컴파일되지 않을 것이다. 왜 typedef를 사용하면 되고, #define을 사용하면 안 되는 지 설명하여라.

Sol)

Typedef의 경우

```
[romance@161s ch9_code]$ ./a.out
DOLLARS = 107.00
[romance@161s ch9_code]$
```

#define의 경우

```
[romance@161s ch9_code]$ cc 9_2.c
9_2.c: In function `main':
9_2.c:7: warning: useless keyword or type name in empty declaration
9_2.c:7: warning: empty declaration
9_2.c:8: parse error before `='
9_2.c:9: parse error before `float'
[romance@161s ch9_code]$ cc -E 9_2.c
# 2 "9_2.c"
int main(void)
{
    float amount = 100.00,
                interest = 0.07 * amount;
  {
    float float;
     float = amount +  interest ;
     printf("DOLLARS = %.2f₩n",float);
```

```
  }
 return 0;
}
[romance@161s ch9_code]$
```

-E 옵션을 사용해 보았을 때, DOLLARS라는 정의한 키워드가 다른 상수로 사용된다면, 이 또한 전처리기에서 모두 바꾸어지기 때문에 float DOLLARS는 float float와 같이 변하게 되고, 이것이 에러를 유발하게 된다.

9-3. 연습문제 2번의 프로그램은 특정사항을 설명하기 위해서 새로운 블록을 사용하여 코드가 자연스럽지 않다. 이 연습문제에서는 typedef를 다시 사용하고, 이번에는 코드가 매우 단순하다.

```
- typedef char * string;
- int main(void)
- {
-    string   a[] = {“I”,”like”,”to”,”fight,”}, b[] = {“pinch,”,”and”,”bight.”};
-  printf(“%s %s %s %s %s %s₩n”,a[0].a[1].a[2].a[3].b[0],b[1].b[2]);
- return 0;
- }
```

먼저 이프로그램을 실행시켜라.

```
[romance@161s ch9_code]$ ./a.out
I like to fight, pinch, and bight.
[romance@161s ch9_code]$
```

이 프로그램에서 typedef를 다음으로 대치하여라.

#define string char *

이제 컴파일이 되지 않을 것이다. 왜 그런가?

char *a[] … ,b[]로 걸리기 때문이다.
따라서 b[]앞에 *를 붙여주면 된다.

9-4. 행렬 a와 b를 더해 c에 저장하는 함수 add(a,b,c)를 작성하고, 이 함수를 사용하는 프로그램을 작성하여 수행시켜 보아라.

Sol)

C code

```
#include <stdio.h>
#define X 3
#define Y 3
void add(int a[][Y],int b[][Y],int c[][Y]);
int main(void)
{ int i,j;
int a[X][Y] = {{1,1,1},{1,0,1},{1,1,1}};
int b[X][Y] = {{2,2,2},{2,0,2},{1,1,1}};
int c[X][Y] = {{0,0,0},{0,0,0},{0,0,0}};
add(a,b,c);
for (i=0; i<X; ++i)
 for (j=0; j<Y; ++j)
    printf("[%d][%d] == %d ",i,j,c[i][j]);
return 0;
}

void add(int a[][Y],int b[][Y],int c[][Y])
{ int i,j;
  for (i=0; i<X; ++i)
    for (j=0; j<Y; ++j)
      c[i][j] = a[i][j] +  b[i][j];
}
```

Result (결과)

```
[romance@161s ch9_code]$ ./a.out
  [0][0] == 3     [0][1] == 3     [0][2] == 3
  [1][0] == 3     [1][1] == 0     [1][2] == 3
  [2][0] == 2     [2][1] == 2     [2][2] == 2
  [romance@161s ch9_code]$
```

9-6. 다음 프로그램은 “double 형의 인자를 하나 갖고 double 형의 값을 리턴하는 함수의 포인터” 형을 명명하기 위해 typedef를 사용한다. 먼자 다음 프로그램을 수행해 보아라.

```
- #include <stdio.h>
- #include <math.h>
-
- # define PI 3.14159
- typedef double dbl;
- typedef dbl (*PFDD)(dbl);
-
- int main(void)
- {
-    PFDD f = sin,g = cos;
-    printf(“f(%f) = %f₩n”,PI,f(PI));
-    printf(“g(%f) = %f₩n”,PI,g(PI));
-  return 0;
- }
```

이 프로그램의 두 번째 typedef를 다음으로 대치하여라.

```
- typedef dbl FDD(dbl);
- typedef FDD *PFDD;
```

이때도 프로그램이 실행되는가? PFDD에 대한 typedef가 어떻게 동작하는지를 설명하여라.

Sol)

```
[romance@161s ch9_code]$ ./a.out
f(3.141590) = 0.000003
g(3.141590) = -1.000000
[romance@161s ch9_code]$
```

9-7. 복소수 뺄셈을 수행하는 함수를 작성하여라. 두 버전을 작성하는데 한 함수는 complex 형의 값을 리턴하도록 하여라.

Sol)

C code

```
#include <stdio.h>
typedef struct _complex { float im;   float re; } complex;
complex minComplex(complex ,complex );
int main(void)
{
complex af,b,c;
b.re = 4; b.im = 5;
c.re = 2; c.im = 3;
af = minComplex(b,c);
printf("%f , %f ₩n",af.re,af.im);
return 0;
}

complex minComplex(complex b,complex c)
{
  complex a;
  a.re = b.re - c.re;
  a.im = b.im - c.im;
  return a;
}
```

Result (결과)

```
[romance@161s ch9_code]$ ./a.out
2.000000 , 2.000000
[romance@161s ch9_code]$
```

9-11. data 파일에 학생 이름, 학번, 성적 등의 목록을 저장하여라. 예를 들면 다음과 같은 형태로 저장한다.

- Casanova 910017 A
- Smith 934422 C
- Jones 878766 B

이 파일에서 자료를 읽어 struct student 형의 배열에 저장하는 recoder라는 프로그램을 작성하여라. 프로그램의 수행은 다음과 같은 재지정 명령을 사용하면 된다.

Reorder < datat

이 프로그램은 또한 성적순으로 학생 이름과 성적을 출력하는 기능도 가지고 있어야 한다. 성적순으로 출력할 때 A학점 받은 학생들을 먼저 출력하고, 그 다음 B 학점순으로 출력한다. 같은 학점을 받은 학생들은 이름순으로 출력한다.

Sol)