**Report**

**For C Programming**

**학과 :**

**학번 :**

**이름 :**

컴퓨터공학과

20184071

김도현

**목차**

1) 정수형과 상수형 변수에 대해 실습하기

┕ 실습 목표

┕ 프로그램 소스코드 및 설명 및 결과

2) 부동 소수점형 변수에 대하여 실습해보기

┕ 실습 목표

┕ 프로그램 소스코드 및 설명과 결과

3) 사용자에게 실수를 입력 받고, 소수점 표기 및 지수 표기 방법으로 출력하기

┕ 실습 목표

┕ 프로그램 소스코드 및 설명과 결과

4) 기호 상수를 사용하여 미터를 평방미터로 변환하기

┕ 실습 목표

┕ 프로그램 소스코드 및 설명과 결과

5) 사용자로부터 실수를 입력 받은 후 다항식을 계산하기

┕ 실습 목표

┕ 프로그램 소스코드 및 설명과 결과

**정수형과 상수형 변수**

1) 실습 목표

┕ 실수의 저장 타입인 short 형의 변수를 선언하고 오버플로우를 확인한다. 또한 형식지정자를 사용하여 8진수 16 진수로 출력한다. 마지막으로 short 형이 아닌 unsigned short 형으로 바꿔서 출력해본다.

2) -0 (실습 0) 프로그램 소스코드 및 설명과 결과

#include <stdio.h>

int main() {

// 실습 0

short i; //정수를 담는 변수인 short 타입의 변수를 선언한다.

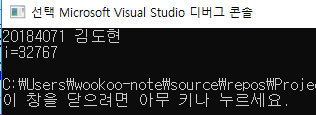
i = 32767; //i 에다 32767 을 담는다

printf("20184071 김도현\n");

printf("i=%d\n", i); // 정수형의 형식 지정자인 %d를 사용하여 i 를 출력한다.

return 0; //main 함수가 잘 작동되었기에 0을 리턴한다.

}



실행결과 ) :Short 형이 최대로 담을 수 있는 수보다 적은 수기 때문에 문제없이 실행된다.

2) -1 (실습 1) 프로그램 소스코드 및 설명과 결과

#include <stdio.h>

int main() {

// 실습 1 - 1

short i; //정수를 담는 변수인 short 타입의 변수를 선언한다.

i = 32767; //i 에다 32767 을 담는다

printf("20184071 김도현\n");

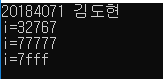
printf("i=%d\n", i); // 정수형의 형식 지정자를 사용하여 i 를 출력한다.

printf("i=%o\n", i);// 8진수형의 형식 지정자를 사용하여 i 를 출력한다.

printf("i=%x\n", i);// 16진수형의 형식 지정자를 사용하여 i 를 출력한다.

return 0; //main 함수가 잘 작동되었기에 0을 리턴한다.

}



실행결과 ) :32767 이 10/8/16 진수로 정상적으로 출력된다.

2) -2 (실습 2) 프로그램 소스코드 및 설명과 결과

#include <stdio.h>

int main() {

// 실습 1 - 2

short i; //정수를 담는 변수인 short 타입의 변수를 선언한다.

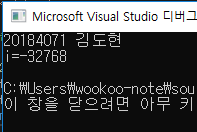
i = 32768; //i 에다 32768 을 담는다.(short 형이 담을수 있는 수보다 큰수)

printf("20184071 김도현\n");

printf("i=%d\n", i); // 정수형의 형식 지정자인 %d를 사용하여 i 를 출력한다.

return 0; //main 함수가 잘 작동되었기에 0을 리턴한다.

}



실행결과 ) :Short 형이 최대로 담을 수 있는 수보다 큰 수를 넣었기 때문에 오버 플로우가 일어난다.

2) -3 (실습 3) 프로그램 소스코드 및 설명과 결과

#include <stdio.h>

int main() {

// 실습 1 - 3

unsigned short i; //정수를 담는 변수인 short 타입의 변수를 선언한다.

//short 형에 unsigned 를 붙여서 음수를 담는 메모리를 양수로 끌어온다.

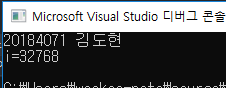
i = 32768; //i 에다 32768 을 담는다

printf("20184071 김도현\n");

printf("i=%d\n", i); // 정수형의 형식 지정자를 사용하여 i 를 출력한다.

return 0; //main 함수가 잘 작동되었기에 0을 리턴한다.

}



실행결과 ) :Short 형이 음수를 담는 메모리를 unsigned 예약어를 통해 양수에 할당하여 정상적으로 저장을 가능하게 한 후 출력을 하니 정상적으로 출력된다.

**부동 소수점 변수에 대해 실습**

1) 실습 목표

┕ 부동소수점의 변수를 생성하고, 오버플로우 및 언더플로우를 알아본다.

2) -1 (실습 1) 프로그램 소스코드 및 설명과 결과

#include <stdio.h>

int main() {

//실습 2 - 1

float x; //실수형 변수 x 를 선언한다

x = 1.2345678901234567890; // 실수형 변수에 1.2345678901234567890를 넣는다

printf("20184071 김도현\n");

printf("x = %10.20f\n", x); //실수형 형식지정자에 정수부분 10자리

//소수부분 20자리까지 출력되게끔 명령한 후 x 를 출력한다.

return 0;

}



실행결과 ) : 1.2345678901234567890 로 선언한 x 가 정상적으로 출력되지 않고 데이터가 조금 깨져서 나온다. 원인을 검색해 보니, 컴퓨터가 부동소수점을 저장할 때, 생기는 구조적 문제이기 때문이라고 한다.

2) -2 (실습 2) 프로그램 소스코드 및 설명과 결과

#include <stdio.h>

int main() {

//실습 2 – 2

float x; //실수형 변수 x 를 선언한다

x = 1e39; // 실수형 변수에 1e39를 넣는다

printf("20184071 김도현\n");

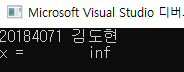
printf("x = %10.20f\n", x); /\*실수형 형식지정자에

정수부분 10자리 소수부분 20자리까지 출력되게끔

명령한 후 x 를 출력한다.\*/

return 0;

}



실행결과 ) :1e39 로 선언한 x 가 정상적으로 출력되지 않고 inf 로 출력된다. Inf 는 무한대로써, float 이 저장할수 있는 값 보다 큰 값을 저장하라고 선언했기 때문이다.

실행결과 보고 요약한다면, 오버 플로우가 일어났음을 알 수 있다.

2) -3 (실습 3) 프로그램 소스코드 및 설명과 결과

#include <stdio.h>

int main() {

//실습 2 - 3

float x; //실수형 변수 x 를 선언한다

x = 1.23456e-46; // 실수형 변수에 1.23456e-46를 넣는다

printf("20184071 김도현\n");

printf("x = %10.20f\n", x); /\*

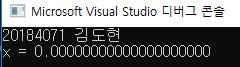
실수형 형식지정자에 정수부분

10자리 소수부분 20자리까지 출력되게끔

한 후 x 를 출력한다.\*/

return 0;

}



실행결과 ) :1.23456e-46으로 선언한 x 가 정상적으로 출력되지 않고 0.0000… 으로 나온다. 이는float 형이 저장할 수 있는 최소 값보다 작은 값을 넣었기 때문에 0이 나온다.

실행결과 보고 요약한다면, 언더플로우 일어났음을 알 수 있다.

**입력받은 실수를 소수점 및 지수표기 방식으로 출력**

1) 실습 목표

┕ 부동소수점의 변수를 생성하고 부동소수점 변수를 출력하고, 실수 형식으로 출력해보고, 지수 형식으로 출력 함으로써 형식지정자에 익숙해진다.

2) -1 (실습 1) 프로그램 소스코드 및 설명과 결과

#include <stdio.h>

int main() {

float input; //float 형식의 변수를 선언한다

printf("실수를 입력하세요 : ");

// 사용자에게 받은 값을 부동소수점으로 저장하고 input 주소에 저장한다

scanf("%f", &input);

//%f 를 사용하여 실수형 그대로 출력한다

printf("실수 형식으로는 %f 이고\n", input);

//%e 를 사용하여 지수형으로 바꾼후 출력한다.

printf("지수 형식으로는 %e 입니다.", input);

return 0;

}



실행결과 ) : 89.567 로 사용자에게 입력받은 실수가 정상적으로 질수형 및 지수 형으로 출력이 됨을 알 수 있다.

**기호 상수를 사용하여 미터를 평방미터로 변환**

1) 실습 목표

┕ 정수형의 변수를 선언하고 사용자에게 입력 받은 후, 기호 상수를 정의 한 후, 기호상수와 사용자에게 입력 받은 값을 곱하여 출력하는 프로그램을 작성한다.

프로그램을 작성하면서 기호 상수를 어떻게 선언하는지 보고, 사용하여 기호상수의 개념을 익힌다.

2) -1 (실습 1) 프로그램 소스코드 및 설명과 결과

#include <stdio.h>

int main() {

int input; //사용자에게 입력 받을 값을 선언해준다.

double results; //결과값을 저장할 변수를 선언해준다.

const double SQMETER\_PER\_PYEONG = 3.3058; //더블 형식의 기호상수를 선언한다.

printf("평을 입력하세요 : ");

scanf("%d", &input);//사용자에게 입력 받은 값을 정수형으로 input 주소에 넣는다.

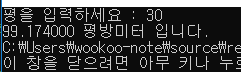
//결과는 3.3058인 기호상수와 input 을 곱해야 한다.

results = SQMETER\_PER\_PYEONG \* input;

printf("%f 평방미터 입니다.", results);

return 0;

}



실행결과 ) : 30 이 정상적으로 3.3 과 곱해져 나옴을 알 수 있다.

SQMETER\_PER\_PYEONG 는 기호 상수로써 const 예약어는 자바에선 final 과 같은 역할을 한다. 이는 변수가 아닌 ‘상수’ 이므로 사용자가 수정할 수가 없다.

**기호 상수를 사용하여 미터를 평방미터로 변환**

1) 실습 목표

┕ 정수형의 변수를 선언하고 사용자에게 입력 받은 후, 기호 상수를 정의 한 후, 기호상수와 사용자에게 입력 받은 값을 곱하여 출력하는 프로그램을 작성한다.

프로그램을 작성하면서 기호 상수를 어떻게 선언하는지 보고, 사용하여 기호상수의 개념을 익힌다.

2) -1 (실습 1) 프로그램 소스코드 및 설명과 결과

#include <stdio.h>

int main() {

float x; //사용자에게 입력받을 값을 선언해줍니다

printf("x의 값을 입력하세요. : ");

//%f 형식 지정자를 사용하여 사용자에게 입력 받은 값을 실수로 x 주소에 넣습니다

scanf("%f", &x);

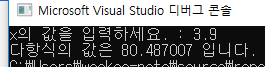
//결과 값은 실수이므로 %f 를 사용하여 출력하게 합니다.

//제곱은 변수를 n 번 곱해주는것이기 때문에 x 를 3번 2번 곱해줍니다.

printf("다항식의 값은 %f 입니다.", ((3 \* x\* x \*x) - (7 \* x\* x) + 9));

return 0;

}



실행결과 ) : 3x^3 – 7x^2 + 9 의 값이 x 가 3.9 일때 정상적으로 출력됩니다.