

포토폴리오

20160742

박지환

(프로그래밍 원리)

10 10

프로그래밍 3차원 5단계

프로그래밍 3차원 5단계 → 소스 편집 → 컴파일 → 링크 → 실행

소스

(프로그래밍 언어로 만든 원본의 프로그램)

→ 기밀된 파일: 소스 파일

컴파일러

프로그래밍 언어로 작성된 소스파일에서 기계어로 작성된 목적 파일을 만들어 내는 프로그램

→ 기계어로 번역된 프로그램: 목적코드

링크

하나 이상의 목적 파일을 하나의 실행 파일로 만들어 주는 프로그램

→ 이러한 과정을 '링크(link)' 또는 '링크(linking)'이라고 함

라이브러리

프로그램을 작성할 때 미리 작성된 프로그램이 미리 만들어

컴파일러 저장

오류와 디버깅

1) 오류

프로그램 개발 과정에서 (타이핑) 모든 문제를 막아야

2) 디버깅

개발과정에서 발생하는 오류를 찾아 소스를 수정하여 다시 컴파일, 링크 실행하는 과정

IDE

프로그램 개발에 필요한 편집기, 컴파일러, 링커, 디버깅을 통합하여 제공하는 통합개발환경

morning

02 03 C프로그래밍 이해와 디버깅과정

프로그래밍의 이해

하스

C프로그래밍과 같은 절차지향 프로그래밍

1. 사용자 정의 함수

프로그래머가 직접 판독 함수

2. 라이브러리 함수

시스템이 미리 만들어 놓은 함수

020 프로그램기초

프로그램 기초

(프로그램은 국어와 maths 등을 하는 컴퓨터에 응용 프로그램은 그 실행)

메이커()

실행되면 가장 먼저 시작되는 부분

기초

문법적으로 교육한 이후에 같은 예약된 단어

예약어

프로그램에 자기 이름을 정의해서 사용되는 단어

작성 규칙

- 1) 숫자는 맨 앞에 올 수 없다.
- 2) 대문자는 구별된다
- 3) 줄 안에 공백문자가 들어갈 수 없다
- 4) 키워드는 식별자로 사용할 수 없다.
- 5) 알파벳과 -를 제외한 문자를 사용할 수 없다.

문법과 규칙

문장

컴퓨터에게 명령을 내리는 최소한의

구문

관한 문법과 같은 자신을 위해서 다른 문법과 함께

모든 사람이 이해할 수 있도록 도움.

2) 자료형 변수 지칭

자료형

1. 기본형, 유한형, 사용자정의형

↳ 정수형, 실수형, 문자형, Void

변수 선언과 초기화

오버로드와 연어 부호
치호형 방법에서 (보이낸) 자장 (오버로드) = 연어 부호가 붙는다.

④ 상수 이름과 포인터 방법

상수
자바 상수, 상수 상수

연어 상수
기타는 enum로 사용하며 상수 상수 상수 상수 상수 상수

아고 상수
define에 대한 상수 상수 상수 상수 상수 상수

4 장 2 예

전역 변수

변수는 함수가 호출될 때 기억에 할당되는 지역 변수와 프로그램 실행이 끝날 때까지 존재하는 전역 변수가 있다.

전역 변수

include

정식 파일

함수 정의와 인터프리터, 컴파일러를 구별

참고

scanf()

05 연산식

연산식

변수에 값을 할당하는 연산식을 포함한다.

연산자

산술 연산자 +, -, *, / 등과 같이 이미 정의된 연산자를 수행하는

연산자 이항 연산자 기호

발행, 이항, 삼항 연산자

++ 연산자가 앞에 있으면 전위 연산자 ++ 연산자가 뒤에 있으면 후위 연산자

순회변수와
은은과 11은 왼쪽 피연산자만은 등기 연산
오른쪽 피연산자도 평가되지 않는다. 값이 저장

비동기 연산자
← →

명칭자
is.off

제어문자열

제어문의 종류

순차문, 선택문

프로그램의 실행 순서 결정

조건문

if ...

반복

for ...

while

break ...

switch 문

다중 선택 구문인 switch 문 사용

6/5/2008

문제는 7가지 있는 것

{int M = 20;

int *p = &M;

*p = 200;

→ 4byte

안됨.

바일드 변수 참조 (q, r, s = {2, 3, 4, 5})

원래: {1, 2}

= *p(i)

q {2 | 3 | 4 | 5}

int q = 5;

int *p = q;

*p = 5;

*p = 6;

p[2] = 7;

→ 배열 참조

문제는

{int *q = q+1};

*q = 10;

→ 3배

(q[0] q[1] q[2] q[3])

= *q[0] *q[1] *q[2] *q[3]

*q

차원 배열

23+9월 배양

영

다들 물어! 원 500. 1000 정도

$$4 \times 10 \times 10 = 10^2$$

조언해 내어

2차원 배열 포인터

in $Q=5, f=10, C=20$

```
int (*p)() { return 4; }
```

$$(n) \neq \{a, b, c\}$$

25-5-20

$$K(\mathbb{Q}) = S_1$$

mg 62

前 $\sqrt{37}$

반복 지점
반복, 수렴

비율공 공제
10.10.10
10.10.10

포인터 변수와 연산

주소
메모리 공간은 8비트인 1바이트마다 순차적인 지점번호를
주소 변수
주소

포인터 변수

주소값 저장하는 변수

선언 방법

포인터 변수 선언에서 지점명과 변수 이름 사이에 *를

관습의 개념

관습

프로그램에서 여러번 등장한 개념을 습관화하도록 학습하기
습관된 프로그램 이해

전자책 프로그램

관습 관습의 프로그램 이해

프로그램 프로그램

프로그램 프로그램

관습점의

관습 점의

관습점과 관습점 - 프로그램

습관 문제

관습 관습은 관습은 return 관습은 관습

관습 관습

관습 관습은 관습은 관습

address.c

```
#define _CRT_SECURE_NO_WARNINGS
```

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    int input;
```

```
    printf("정수 입력: ");
```

```
    scanf("%d", &input);
```

```
    printf("입력 값: %d\n", input);
```

```
    printf("주소 값: %u(10진수), %p(16진수)\n", (int) &input, &input);
```

```
    printf("주소 값: %d(10진수), %#X(16진수)\n", (unsigned) &input,  
(int) &input);
```

```
    printf("주소 값 크기: %d\n", sizeof(&input));
```

```
    return 0;
```

```
}
```

basicpointer.c

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    char c = '@';
```

```
    char *pc = &c;
```

```
    int m = 100;
```

```
    int *pm = &m;
```

```
    double x = 5.83;
```

```
    double *px = &x;
```

```
    printf("변수명   주소값   저장값\n");
```

```
    printf("-----\n");
```

```
    printf("%3s %12p %9c\n", "c", pc, c);
```

```
    printf("%3s %12p %9d\n", "m", pm, m);
```

```
    printf("%3s %12p %9f\n", "x", px, x);
```

```
    return 0;
```

```
}
```


swap.c

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    int m = 100, n = 200, dummy;
```

```
    printf("%d %d\n", m, n);
```

```
    //변수 m과 n을 사용하지 않고 두 변수를 서로 교환
```

```
    int *p = &m; //포인터 p가 m을 가리키도록
```

```
    dummy = *p; //변수 dummy에 m을 저장
```

```
    *p = n;           //변수 m에 n을 저장
```

```
    p = &n;          //포인터 p가 n을 가리키도록
```

```
    *p = dummy; //변수 n에 dummy 값 저장
```

```
    printf("%d %d\n", m, n);
```

```
    return 0;
```

```
}
```

```
array.c
#include <stdio.h>
#define SIZE 3

int main(void)
{
    int score[] = { 89, 98, 76 };

    //배열이름 score는 첫 번째 원소의 주소
    printf("score: %p, &score[0]: %p\n", score, &score[0]);

    //배열이름 score는 첫 번째 값
    printf("*score: %d, score[0]: %d\n\n", *score, score[0]);

    printf("첨자   주소       저장값\n");
    //배열이름 score를 사용한 주소와 원소 값 참조
    for (int i = 0; i < SIZE; i++)
        printf("%2d %10p %6d\n", i, (score + i), *(score + i));

    return 0;
}
```

arraysize.c

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    int data[] = {3, 4, 5, 7, 9};
```

```
    printf("%d %d\n", sizeof(data), sizeof(data[0]));
```

```
    printf("일차원 배열: 배열 크기 == %d\n", sizeof(data) /  
sizeof(data[0]));
```

```
    //4 x 3 행렬
```

```
    double x[][3] = { { 1, 2, 3 }, { 7, 8, 9 }, { 4, 5, 6 }, { 10, 11, 12  
} };
```

```
    printf("%d %d %d\n", sizeof(x), sizeof(x[0]), sizeof(x[1]),  
sizeof(x[0][0]));
```

```
    int rowsize = sizeof(x) / sizeof(x[0]);
```

```
    int colsize = sizeof(x[0]) / sizeof(x[0][0]);
```

```
    printf("이차원 배열: 행수 == %d 열수 == %d\n", rowsize, colsize);
```

```
    printf("이차원 배열: 전체 원소 수 == %d\n", sizeof(x) /  
sizeof(x[0][0]));
```

```
    return 0;
```

```
}
```

```
inputarray.c
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    //초기화로 모든 원소에 0을 저장
    int input[20] = { 0 };

    printf("배열에 저장할 정수를 여러 개 입력하십시오.");
    printf(" 0을 입력하면 입력을 종료합니다.\n");
    int i = 0;
    do {
        scanf("%d", &input[i]);
    } while (input[i++] != 0);

    i = 0;
    while (input[i] != 0) {
        printf("%d ", input[i++]);
    }
    puts("");

    return 0;
}
```

doubletoint.c

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    double dint[2] = { 0.0 };
```

```
    int *p = (int *) dint;
```

```
    p[0] = 1;
```

```
    p[1] = 2;
```

```
    p[2] = 3;
```

```
    p[3] = 4;
```

```
    for (int i = 0; i < 4; i++)
```

```
        printf("%p %d\n", p+i, *(p + i));
```

```
    return 0;
```

```
}
```


constptr.c

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int i = 10, j = 20;
```

```
    const int *p = &i; /*p가 상수로 *p로 수정할 수 없음
```

```
    // *p = 20; //오류 발생
```

```
    p = &j;
```

```
    printf("%d\n", *p);
```

```
    double d = 7.8, e = 2.7;
```

```
    double * const pd = &d;
```

```
    //pd = &e; //pd가 상수로 다른 주소 값을 저장할 수 없음
```

```
    *pd = 4.4;
```

```
    printf("%f\n", *pd);
```

```
    return 0;
```

```
}
```

pointer.c

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    int data = 100;
```

```
    int *ptring;
```

```
    ptring = &data;
```

```
    printf("변수명   주소값           저장값\n");
```

```
    printf("-----\n");
```

```
    printf("  data   %p   %8d\n", &data, data);
```

```
    printf("ptring  %p   %p\n", &ptring, ptring);
```

```
    return 0;
```

```
}
```

compoundassign.c

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    int x = 5, y = 10;

    printf("두 정수를 입력 >> ", &x, &y);
    scanf("%d%d", &x, &y);

    printf("The addition is: %d\n", x += y);
    printf("x = %d, y = %d\n", x, y);
    printf("The subtraction is: %d\n", x -= y);
    printf("x = %d, y = %d\n", x, y);
    printf("The multiplication is: %d\n", x *= y);
    printf("x = %d, y = %d\n", x, y);
    printf("The division is: %d\n", x /= y);
    printf("x = %d, y = %d\n", x, y);
    printf("The remainder is: %d\n", x %= y);
    printf("x = %d, y = %d\n", x, y);
    printf("x *= x + y is: %d\n", x *= x + y);
    printf("x = %d, y = %d\n", x, y);

    return 0;
}
```

char.c

#define _CRT_SECURE_NO_WARNINGS //scanf() 오류를 방지하기 위한

상수 정의

#include <stdio.h>

#include <ctype.h> //문자 관련 함수는 헤더파일 ctype.h에 매크로로 정의

void print2char(char);

int main(void)

{

 char ch;

 printf("알파벳(종료x) 또는 다른 문자 입력하세요.\n");

 do

 {

 printf("문자 입력 후 Enter: ");

 scanf("%c", &ch);

 getchar(); //enter 키 입력 받음

 if (isalpha(ch))

 print2char(ch);

 else

 printf("입력: %c\n", ch);

 } while (ch != 'x' && ch != 'X'); //입력이 x 또는 X이면 종료

 return 0;

}

void print2char(char ch)

{

 if (isupper(ch))

 printf("입력: %c, 변환: %c\n", ch, tolower(ch));

 else

 printf("입력: %c, 변환: %c\n", ch, toupper(ch));

 return;

}

factorial.c

```
#include <stdio.h>
```

```
int factorial(int); //함수원형
```

```
int main(void)
```

```
{
```

```
    for (int i = 1; i <= 10; i++)
```

```
        printf("%2d! = %d\n", i, factorial(i));
```

```
    return 0;
```

```
}
```

```
// n! 구하는 재귀함수
```

```
int factorial(int number)
```

```
{
```

```
    if (number <= 1)
```

```
        return 1;
```

```
    else
```

```
        return (number * factorial(number - 1));
```

```
}
```


if.c

```
#define _CRT_SECURE_NO_WARNINGS
```

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    double temperature;
```

```
    printf("현재 온도 입력: ");
```

```
    scanf("%lf", &temperature);
```

```
    if (temperature >= 32.0)
```

```
    {
```

```
        printf("폭염 주의보를 발령합니다.\n");
```

```
        printf("건강에 유의하세요.\n");
```

```
    }
```

```
    printf("현재 온도는 섭씨 %.2f 입니다.\n", temperature);
```

```
    return 0;
```

```
}
```

prj1-04.c

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    int row = 10;
```

```
    int out = 2;
```

```
    for (int i = 0; i < row; i++)
```

```
    {
```

```
        int c;
```

```
        for (c = 0; c <= (row - i); c++)
```

```
            printf(" ");
```

```
        int devider;
```

```
        for (c = 0; c <= i; c++)
```

```
        {
```

```
            while (1) {
```

```
                for (devider = 2; devider <= out - 1;
```

```
devider++)
```

```
                {
```

```
                    if (out % devider == 0)
```

```
                        break;
```

```
                }
```

```
                if (out++ == devider)
```

```
                {
```

```
                    printf("%3d ", out-1);
```

```
                    break;
```

```
                }
```

```
            }
```

```
        }
```

```
        printf("\n");
```

```
    }
```

```
}
```