

함수 포인터

- 함수 포인터, void 포인터, main 함수로의 인자 전달 -

성공회대학교 IT융합자율학부
소프트웨어공학전공
홍 성 준

함수 포인터

◎ 함수 포인터

- 메모리에 저장된 함수의 주솟값을 저장하는 포인터
 - 프로그램 실행의 흐름을 구성하는 함수도 바이너리 형태로 '메인 메모리' 공간에 저장

◎ 함수 이름

- 함수가 저장된 메모리 공간을 가리키는 함수 포인터 상수

함수 포인터

◎ 함수 포인터 변수

- 메모리에 저장된 함수의 주솟값을 저장하기 위한 포인터 변수
- 함수 포인터 형
 - 함수의 반환형과 매개변수 선언 형태에 대한 정보를 가짐
 - 함수의 반환형과 매개변수 선언이 같은 함수의 함수 포인터 형은 일치함

`int SimpleFunc(int num)` 반환형 **int**, 매개변수 **int형 1개**

`double ComplexFunc(double num1, double num2)` 반환형 **double**, 매개변수 **double형 2개**

- 반환형이 int이고 매개변수로 int형 변수가 1개 선언된 포인터 형
- 반환형이 double이고 매개변수로 double형 변수가 2개 선언된 포인터 형

함수 포인터

◎ 함수 포인터 변수 선언과 사용

```
int (*fptr) (int)
```

fptr은 포인터!

```
int (*fptr) (int)
```

반환형이 int인 함수 포인터!

```
int (*fptr) (int)
```

매개변수 선언이 int 하나인 함수 포인터!

```
int SoSimple(int num1, int num2) { . . . }  
int (*fptr) (int, int); // SoSimple 함수이름과 동일한 형의 변수 선언  
fptr=SoSimple; // 상수의 값을 변수에 저장  
fptr(3, 4); // SoSimple(3, 4)와 동일한 결과를 보임
```

- fptr과 SoSimple에 같은 값이 저장되어 있어, fptr을 통해 SoSimple 함수를 호출할 수 있음
- 함수 포인터 변수나 함수 포인터 상수냐의 차이

함수 포인터

© FunctionPointer.c

```
void SimpleAdder(int n1, int n2)
{
    printf("%d + %d = %d \n", n1, n2, n1+n2);
}

void ShowString(char * str)
{
    printf("%s \n", str);
}

int main(void)
{
    char * str="Function Pointer";
    int num1=10, num2=20;

    void (*fptr1)(int, int) = SimpleAdder;
    void (*fptr2)(char *) = ShowString;

    /* 함수 포인터 변수에 의한 호출 */
    fptr1(num1, num2);
    fptr2(str);
    return 0;
}
```

10 + 20 = 30
Function Pointer

함수 포인터

© UsefulFunctionPointer.c

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

```
int WhoIsFirst(int age1, int age2, int (*cmp)(int n1, int n2))
{
    return cmp(age1, age2);
}
```

```
int OlderFirst(int age1, int age2)
{
    if(age1>age2)
        return age1;
    else if(age1<age2)
        return age2;
    else
        return 0;
}
```

- 함수 WhoIsFirst의 세번째 인자로 어떤 함수의 주소값이 전달되느냐에 따라 함수 WhoIsFirst의 동작 방식이 결정 됨

```
int YoungerFirst(int age1, int age2)
{
    if(age1<age2)
        return age1;
    else if(age1>age2)
        return age2;
    else
        return 0;
}
```

```
int main(void)
{
    int age1=20;
    int age2=30;
    int first;

    printf("입장순서 1 \n");
    first=WhoIsFirst(age1, age2, OlderFirst);
    printf("%d세와 %d세 중 %d세가 먼저 입장! \n\n", age1, age2, first);

    printf("입장순서 2 \n");
    first=WhoIsFirst(age1, age2, YoungerFirst);
    printf("%d세와 %d세 중 %d세가 먼저 입장! \n\n", age1, age2, first);
    return 0;
}
```

void형 포인터

◎ void형 포인터 변수

- 형(type)이 존재하지 않는 포인터 변수
 - 형 정보가 존재하지 않기 때문에 어떤 주소값도 저장이 가능
 - 형 정보가 존재하지 않기 때문에 포인터 연산이나 값의 변경이나 참조를 위한 * 연산자 사용 불가

```
void * ptr;
```

```
int main(void)
{
    int num=20;
    void * ptr=&num;
    *ptr=20;    // 컴파일 에러!
    ptr++;     // 컴파일 에러!
    . . . . .
}
```

main 함수로의 인자 전달

◎ main 함수를 통한 인자 전달

- 프로그램 실행 시 main 함수로 전달할 인자를 열거할 수 있음

```
C:\WINDOWS\system32\cmd.exe

C:\VI00001\Hello>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 3664-D7FE

C:\VI00001\Hello 디렉터리

2021-03-08 오전 04:48 <DIR>          .
2021-03-08 오전 04:48 <DIR>          ..
2021-03-08 오전 04:33 <DIR>          Debug
2021-03-08 오전 04:48             84 Hello.c
2021-03-08 오전 04:29          1,430 Hello.sln
2021-03-08 오전 04:33          7,187 Hello.vcxproj
2021-03-08 오전 04:33           981 Hello.vcxproj.filters
2021-03-08 오전 04:29          168 Hello.vcxproj.user
                5개 파일              9,850 바이트
                3개 디렉터리  205,742,333,952 바이트 남음

C:\VI00001\Hello>dir Hello.c
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 3664-D7FE

C:\VI00001\Hello 디렉터리

2021-03-08 오전 04:48             84 Hello.c
                1개 파일              84 바이트
                0개 디렉터리  205,742,333,952 바이트 남음

C:\VI00001\Hello>_
```

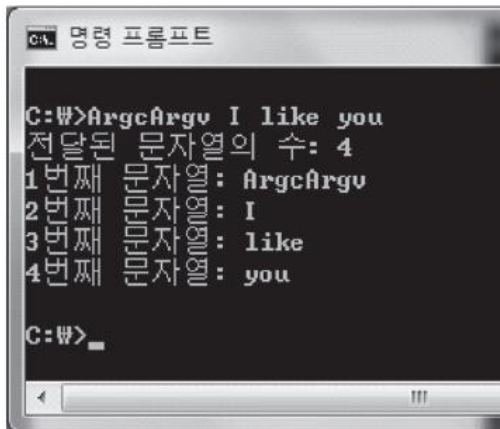

main 함수로의 인자 전달

◎ main 함수를 통한 인자 전달

- ArgcArgv.c

```
int main(int argc, char *argv[])
{
    int i=0;
    printf("전달된 문자열의 수: %d \n", argc);

    for(i=0; i<argc; i++)
        printf("%d번째 문자열: %s \n", i+1, argv[i]);
    return 0;
}
```



```
C:\W>ArgcArgv I like you
전달된 문자열의 수: 4
1번째 문자열: ArgcArgv
2번째 문자열: I
3번째 문자열: like
4번째 문자열: you
C:\W>
```

main 함수로의 인자 전달

◎ char *argv[]

- argv는 char 형 더블 포인터 변수 (char **)
- ArgvParamType.c

```
void ShowAllString(int argc, char * argv[])
{
    int i;
    for(i=0; i<argc; i++)
        printf("%s \n", argv[i]);
}

int main(void)
{
    char * str[3]={
        "C Programming",
        "C++ Programming",
        "JAVA Programming"
    };
    ShowAllString(3, str);
    return 0;
}
```

```
C Programming
C++ Programming
JAVA Programming
```

main 함수로의 인자 전달

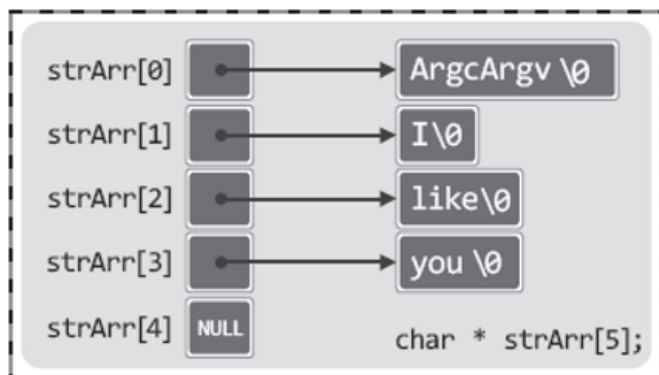
◎ 인자의 형성 과정

c:\>ArgcArgv I like you

문자열의 구분

문자열 1	"ArgcArgv"
문자열 2	"I"
문자열 3	"like"
문자열 4	"you"

문자열의 구성



문자열 기반 함수의 호출

main(4, strArr);

```
int main(int argc, char *argv[])
{
    int i=0;
    printf("전달된 문자열의 수: %d \n", argc);

    while(argv[i]!=NULL)
    {
        printf("%d번째 문자열: %s \n", i+1, argv[i]);
        i++;
    }
    return 0;
}
```

C:\> ArgvEndNULL "I love you"

전달된 문자열의 수: 2

1번째 문자열: ArgvEndNULL

2번째 문자열: I love you