C++ Study

함수 심화/재귀/배열

울산대학교 | 박예찬 | Github:devpyc | yechan6855@naver.com

함수의 오버로딩

오버로딩

같은 이름의 함수를 매개변수의 형태가 다르게 정의하는 것

```
int add(int a, int b){
        return a+b;
}
double add(double a, double b){
        return a+b;
}

cout<<add(3,5)<<” “<<add(3.2,4.5);
```

매개변수 기본값

함수 호출시 매개변수를 넘기지 않으면 기본 값 사용

```
void hello(string s=”Guest”){
    cout<<”Hello “<<s;
}

int main()
     hello(“Tom”);
     hello():
}
```

값 전달과 참조 전달

값 전달

값 전달 방식은 복사본을 전달하여 **원본값이 변하지 않음**

```
void swap(int a,int b){
    int tmp=a;
    a=b;
    b=tmp;
}
```

참조 전달

참조 전달 방식은 원본을 전달하여 **원본값이 변경됨**

```
void swap(int &a,int &b){
    int tmp=a;
    a=b;
    b=tmp;
}
```

재귀 함수

기본 개념

재귀 함수는 **자기 자신을 호출하는 함수**로,
아래의 필수 조건을 충족해야함

1. 종료 조건이 반드시 존재해야함
2. 매번 문제의 크기가 줄어들어야 함

예제 1: 팩토리얼

$$f(n) = n \times f(n-1)$$

```
int factorial(int n){
    if(n==0) return 1;
    return n*factorial(n-1);
}
```

예제 1: 팩토리얼 반복문과의 비교

```
int factorial(int n){
    if(n==0) return 1;
    return n*factorial(n-1);
}

int factorial_loop(int n){
   int ans=0;
   for(int i=1; i<=n; i++) ans*=i;
   return ans;
}
```

예제 2: 피보나치 수

$$f(n) = f(n-1) + f(n-2)$$

```
int fibo(int n){
    if(n<=1) return n;
    return fibo(n-1) + fibo(n-2);
}
```

예제 3: 하노이 탑 이동 횟수

$$f(n) = 2f(n-1) + 1$$

```
int hanoi(int n){
    if(n==0) return 0;
    return 2 * hanoi(n-1)+1;
}
```

배열

정적 배열

컴파일시에 크기가 결정되고, 그 이후 **크기 변경이 불가능**함

```
int arr[5]; // 선언만 함
int arr2[3] = {10, 20, 30}; // 선언과 초기화
```

정적 배열 입력받기

배열의 인덱스는 **0-based**

```
int arr[5];
for(int i=0; i<5; i++){
    cin>>arr[i];
}
```

정적 배열 출력하기

배열의 인덱스는 **0-based**

```
int arr[5];
for(int i=0; i<5; i++){
    cout<<arr[i]<<” “;
}
```

정적 배열 정렬하기

arr은 첫번째 원소의 주소, arr+5는 메모리상에서 5칸 뒤의 주소임

따라서, **주소 범위**를 넘겨줘야 정렬 가능

```
int arr[5]={1,6,2,16,3};
sort(arr,arr+5;)
```

동적 배열

프로그램 **실행중에 크기가 결정**됨

```
int n;
cin>>n;

int* arr = new int[n]; // 동적 할당
for(int i=0; i<n+n; i++){
    cin>>arr[i];
}
delete[] arr;
```

메모리 해제는 필수입니다

동적을 할당 했다면 **반드시 해제**도 합시다!

사용후에 delete[]가 없다면 **메모리 누수** 발생

```
int* arr = new int[n];
```

STL vector

동적 배열: vector

크기 변경도 가능하고 메모리 관리도 자동으로 해줌
제공되는 기능도 유용함

```
vector<int>v;
v.push_back(10);
cout<<v[0]<<”\n”<<v.size();
```

vector 입력받기

```
int n;
cin>>n;
vector<int>v(n);
for (int i=0; i<n; i++){
        cin>>v[i];
}
```

vector 출력하기(1)

Index-based loop

```
for(int i=0; i<n; i++){
    cout<<v[i]<<” “;
}
```

vector 출력하기(2)

Range-based loop

```
for(int i:v){
    cout<<i<<” “;
}
```

vector 정렬하기

첫번째 요소를 가리키는 v**.begin()**,
마지막 요소의 뒤를 가리키는 v**.end()**를 넘겨줌

```
vector<int>v;
sort(v.begin(),v.end());
```

직접 해보기

1. 재귀를 이용하여 피보나치 N번째 수 구하기
2. 정수 5개를 입력받아서 최소/최대 출력하기
3. vector를 이용하여 배열 뒤집기

return 0;

울산대학교 | 박예찬 | Github:devpyc | yechan6855@naver.com