Topic 4 Function
Part 2

Part 2

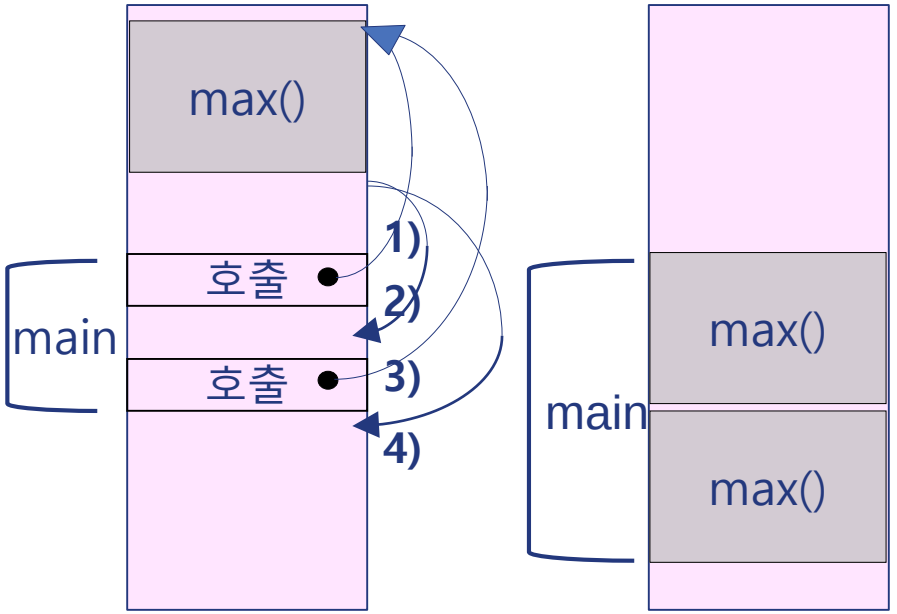
- inline function
- static local variable
- Recursive function
- Pre/post condition
- Block scope vs File scope (internal linkage vs external)
- Function overloading
- Function as argument

inline function

- The call to Inline function is replaced with the actual code instead of function call.
- Meaning of the keyword inline for functions came to mean **"multiple definitions are permitted"** rather than "inlining is preferred" (Since C++17)

```
inline int max(const int i, const int j){
  return ( i > j ) ? i : j ;
}

int main() {
  max(10, 20) ;
  max(30, 20) ;
}
```



https://en.cppreference.com/w/cpp/language/inline

일반함수 vs inline 함수

- non-static 인라인 함수가 서로 다른 파일에 여러 번 선언될 때, 모두 inline 선언해야 하고, 모든 정의는 동일해야 하고, 동일한 주소를 가짐(Since C++17)

```
// MyHeaderForInline.h
#ifndef __MY_HEADER_FOR_INLINE_H
#define __MY_HEADER_FOR_INLINE_H
#include <vector>

void readPositiveNumbers(vector<int>& numbers) ;
void sortNumbers(vector<int>& numbers) ;
void swap(int& n1, int& n2) ;
void printNumbers(const vector<int>& numbers) ;

void swap(int& n1, int& n2) {
    int temp = n1 ;
    n1 = n2 ;
    n2 = temp ;
}
#endif
```

```
// MyHeaderForInline.h
#include <vector>

void readPositiveNumbers(vector<int>& numbers) ;
void sortNumbers(vector<int>& numbers) ;
inline void swap(int& n1, int& n2) ;
void printNumbers(const vector<int>& numbers) ;

inline void swap(int& n1, int& n2) {
    int temp = n1 ;
    n1 = n2 ;
    n2 = temp ;
}
```

```
// main.cpp
#include <iostream>
#include "MyHeaderForInline.h"

int main() {
    int n1=1, n2=2;
    swap(n1, n2);
    cout << n1 << ", " << n2;
}
```

```
// sort_number.cpp
#include "MyHeaderForInline.h"

void sortNumbers(vector<int>& numbers) {
    …
    swap(n1, n2);
    …
}
```

인라인 함수 장단점

- Advantages
  - 함수 호출 오버헤드가 발생하지 않음
  - 함수 호출 시 파라미터 값을 복사하는 오버헤드 없음
  - 컴파일러 최적화를 기대할 수 있음
  - 함수의 명령어 개수가 작으면 임베디드 시스템에 유용함

- Disadvantages
  - 인라인 함수에서 사용하는 변수를 위한 추가적인 레지스터 필요
  - 실행파일의 크기가 증가함 (동일 코드 복제)
  - 명령어 캐시 적중률을 감소시켜 실행 속도가 감소할 수 있음
  - 컴파일 시간을 증가 시킴

https://www.geeksforgeeks.org/inline-functions-cpp/

매크로와 인라인 함수

인라인 함수	매크로

inline int safeABS(int i) {
  return i >= 0 ? i : -i;
}
	#define unsafeABS(i) ((i) >= 0 ? (i) : -(i))

매크로와 인라인 함수

```
# include <iostream>
using namespace std ;

#define unsafeABS(i)  ( (i) >= 0 ? (i) : -(i) )
inline int safeABS(int i) { return i >= 0 ? i : -i; }
int f() {
  static int i = 0 ;
  i-- ;
  return i ;
}
```

인라인 함수 이용시	매크로 이용시

void main() {
  int x = -10 ;
 	
  cout << safeABS(x++) << endl ; // 10
  cout << safeABS(f()) << endl ; // 1
}
	void main() {
	  int x = -10 ;
	
	  cout << unsafeABS(x++) << endl ; // 9
	  cout << unsafeABS(f()) << endl ; // 2
	}