|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2021\_1\_C++\_13\_개념확인 | 학번 : | 20185309 | 이름 : | 황명원 |

* 기본 개념 확인

1. 다음 질문에 대하여 O, X 하시오.
2. 템플릿을 선언하기 위해 사용하는 키워드는 template이나 generic이다 ( x )
3. 제네릭 타입을 선언하는 키워드는 class 이다 ( o )
4. 템플릿 함수와 동일한 이름의 함수가 중복되어 있을 경우 템플릿 함수가 우선적으로 바인딩 된다( x )
5. C++ 표준STL 라이브러리가 작성된 이름공간은 std 이다 ( o )
6. 제네릭 함수나 제네릭 클래스를 활용하는 프로그래밍 기법을 일반화 프로그래밍이라고 한다 ( o )
7. 템플릿 함수는 오버로딩을 할 수 없다( x )
8. 템플릿을 사용하면 컴파일 오류메시지가 풍부하여 디버깅에 많은 도움을 준다 ( x )
9. map 컨테이너 요소들은 키값을 기준으로 내림차순으로 정렬된다 ( x )
10. 람다식에서 리턴타입은 생략할 수 있다 ( o )
11. 람다식에서 함수 바깥의 변수 목록을 사용하려면 캡쳐리스트로 작성하면 된다 ( o )
12. 다음 STL의 각 기능을 사용하기 위해 필요한 헤더 파일을 제시하시오
13. vector 클래스 -> #include <vector>
14. merge 함수 ->#include <algorithm>
15. map 클래스 ->#include <map>
16. 다음에서 템플릿 선언을 잘못한 것은? 2번
17. template <class T>
18. template (class T)
19. template <typename T>
20. template <typename T1, typename T2>
21. 다음 제네릭 함수 선언에서 잘못된 부분을 바르게 수정하시오

template <typename T> int max(T x, T, y){

if(x>y) return x;

else return y;

}

ㅡ수정후ㅡ

template <typename T> int max(T &x, ,T &y){

if(x>y) return x;

else return y;

}

1. 다음 두 함수를 일반화한 제네릭 함수를 작성하시오

void insert(int a, int b[], int index){

b[index] = a;

}

void insert(char a, char \*b, int index){

\*(b+index) = a;

}

ㅡ소스코드ㅡ

template <typename T> void insert(T a,T\* b,int index){  
 \*(b+index) = a;  
}

1. 템플릿 클래스 Container를 작성하고자 한다

template <class T> class Container{

T\* p; //T 타입의 포인터 p를 선언하라

int size; //배열의 크기를 나타내는 변수 size를 선언하라

public:

Container(int n); //멤버 변수 p에 n개의 동적 배열을 할당 받는 생성자

~Container();

void set(int value, T value){ p[index] = value; }

T get(int index); //index 위치의 값 리턴

};

1. 빈칸의 채워 넣으시오

첫번째 빈칸: T\* p;

두번째 빈칸:int size;

1. 생성자를 구현하시오

template <class T> Container<T>::Container(int n){  
 size=n;  
 p=new T[size];  
}

1. 소멸자를 구현하시오

template<class T> Container<T>::~Container() {  
 delete []p;  
}

1. get()을 구현하시오

template<class T> T Container<T>::get(int index) {  
 return index;  
}

1. char 타입의 문자만 저장 가능한 Container 객체 c를 생성하는 선언문을 작성하시오.(c의 크기는 26)

Container<char> c(26);

1. 문제 5에서 생성한 객체 c에 set() 함수를 이용하여 알파벳 ‘a’~’z’를 삽입하고 get() 함수를 이용하여 반대순으로 화면에 출력하는 main() 함수를 작성하시오

int main() {  
 Container<char> c(26);  
 int index=0;  
 c.set(index,'a');  
 for(int i=1; i<26 ; i++)  
 c.set(i,'a'+i);  
 for(int i=25 ; i>=0 ; i--)  
 cout<<c.get(i)<<endl;  
}

1. 제시된 문제를 해결하는 문장을 제시하시오

vector<double> v;

1. 생성된 벡터 v 마지막에 3.1 삽입

v.push\_back(3.1);

1. 벡터 v에 저장된 원소 개수

v.size();

1. 벡터 v에 저장된 첫번째 원소 삭제

vector<double>::iterator it;

it=v.begin();

v.erase(it);

1. vector<char> 타입의 객체 v가 함수 print()의 매개변수로 전달되었을 때 반복문을 이용하여 모든 원소를 화면에 출력하고자 한다. 제시된 방법대로 구현하시오
2. 반복자 사용

void print(vector<int> &v){  
 for (vector<int>::iterator iter = v.begin(); iter != v.end(); ++iter)  
 cout << \*iter << " ";  
}

1. 향상된 for문 사용

void print(const vector<int> &v) {  
 for (int elem : v)  
 cout <<elem<<" ";  
 cout << endl;  
}

1. 매개변수로 배열 크기와 double 타입의 배열을 받아 배열 원소를 역순으로 저장하는 람다식 reverse를 구현하여 auto 변수 method에 저장하고 호출하는 문장을 작성하시오.

void reverse(double a[],int size){  
 int tmp;  
 for(int i=0 ; i<size-1 ; i++){  
 for(int j=0 ;j<size-i-1 ; j++){  
 tmp=a[j];  
 a[j]=a[j+1];  
 a[j+1]=tmp;  
 }  
 }  
}  
int main() {  
  
 auto method=[](double a[],int size){  
 reverse(a,size);  
 for(int i=0 ; i<size ; i++)  
 cout<<a[i]<<" ";  
 };  
  
 double b[5]={1,2,3,4,5};  
 method(b,5);  
  
}

1. C++ STL 구성에 대하여 설명하시오

컨테이너 – 템플릿 클래스

-데이터를 담아두는 자료 구조를 표현한 클래스

-리스트,큐,스택,맵,셋,벡터

iterator – 컨테이너 원소에 대한 포인터

-컨테이너의 원소들을 순회하면서 접근하기 위해 만들어진 컨테이너 원소에 대한 포 인터

알고리즘 – 템플릿 함수

-컨테이너 원소에 대한 복사, 검색, 삭제, 정렬 등의 기능을 구현한 템플릿 함수 – 컨테이너의 멤버 함수 아님