2022 자료구조 중간고사

(1) 시험시간 : 10월 17일(월요일) 오전 **8시 30분 - 10시 20분**
(2) 답안은 제공된 시험지에 작성해야 합니다.
(3) 채점자가 이해하기 쉽도록 코드에는 적절한 설명(comments)을 적으세요.
(4) 필요한 경우 문제지의 뒷면을 사용해도 되지만 표시해야 합니다.
(5) 코드는 명확하게 C++ 문법에 맞도록 작성해야 한다.
(6) Python을 사용할 경우에는 전체 driver 코드를 모두 작성해야 한다.

학번	**이름**	**감독자 확인**

문제 (배점)	점수	문제 (배점)	점수
1 (10)		6 (10)	
2 (10)		7 (20)	
3 (20)		8 (20)	
4 (10)		9 (20)	
5 (10)		10 (20)	
소계		합계 (150)	

[문제1] 모든 코드의 head는 아래가 공통적으로 적용된다.

```
#include <bits/stdc++.h>
#define allout(MSG,X) cout<<"\n"<<MSG<<"=> ";for(auto w:X)cout<<w<<" "
```

Code 1	**stdout**

`void prob1(){`
`  vector< vector<int> > \`
`  myv = {{11,22,8,9}, {33,55,66}, {55, 76}, {-25},{}} ;`
`  cout << "\n Out1 = " << myv[0][1] ;`
`  cout << "\n Out2 = " << myv[2][0] ;`
`  cout << "\n Out3 = " << myv[1].size() + myv.size() ;`
`}`
	Out1 =
	Out2 =
	Out3 =

Code 2	**stdout**

`array< array<int,4>, 3>`
`  X {{{1,2,3,4}, {5,6}, {9,10,11,12}} };`
 	
`  cout << "\n p21 = " << X[2][3] ;`
`  cout << "\n p22 = " << X[1][3] ;`
`  cout << "\n p23 = " << X[1].size() + X.size() ;`
`  int S = 0 ;`
`  for(int i=0 ; i<3; ++i) S += X[i].back() ;`
`  cout << "\n p24 = " << S ;`
	p21 =
	p22 =
	p23 =
	p24 =

Code 3: 일차원 벡터에서 0을 삭제한 후 크기를 재조정하는 작업을 하려고 한다.
아래 결과가 나오도록 박스 속에 문장을 채워 넣으시오. (최대 4개의 문장까지 허용)

```
vector <int> B {4, 0, 0, 3, 6, 0, 7, 0, 0, 9, 0, 1} ;
    [                                                ]
    [                                                ]
    [                                                ]
    [                                                ]
 allout("Prob3 ", B ) ;
```

Prob3 => 4 3 6 7 9 1

Code 4 : 2D vector sorting

```
typedef vector<vector<int>> dvec ;
void prob5(){
    dvec v1 = { {12}     ,{33, 33},{66}, {0,1  },{1} } ;
    dvec v2 = { {12, 22},{33, 32},{67}, {0,1,2},{2} } ;
    dvec v3 = { {11, 22},{ }                    ,{3} } ;
    dvec v4 = { {12, 22},{33}    , {67}         ,{4} } ;
    dvec v5 = { { }      ,{32}                 ,{5} } ;
    dvec v6 = { {12, 22},{ { } } , {66}         ,{6} } ;
    dvec v7 = { {12, 22},{     } , {65}         ,{7} } ;
    vector<dvec> Allvec = { v1, v2, v3, v4, v5, v6, v7 };
    sort( Allvec.begin(), Allvec.end() ) ;
    for( auto w : Allvec ) {
        allout("> ", w.back()) ;
    }
}
```

stdout

```
>
>
>
>
>
>
>
```

Code 5

```
 void prob6(){
    vector <int> Z {1,2,3} ;
    int S = 0, T = 0 ;   // 이 상황에서 Z.capacity() = 3 ;
    for(int i=1; i<=30; i++){
        Z.push_back(i) ;
        T += Z.size( );
        S += Z.capacity( );
    }
    cout << "T= " << T << "\n , S= " << S ;
}
```

stdout

```
T=
S=
```

Code 6

```
bool myfun (int i, int j) {
    if( abs(i-j) <= 1 ) return(1) ;  else return(0) ;
}
prob6() {
      vector<int> myvec1 {15,21,21,20,20,31,20,10,10,11,9,9} ;
      vector<int> myvec2 {15,21,21,20,20,31,20,10,10,11,9,9} ;

      unique (myvec1.begin(), myvec1.end());
      allout("> 1. myvec1[]= ", myvec1 ) ;

      it=unique (myvec2.begin(), myvec2.end(), myfun);
      myvec2.resize( distance(myvec2.begin(),it) );
      allout("> 2. myvec2[]= ", myvec2) ;
}
```

```
> 1. myvec1[]=
> 2. myvec2[]=
```

Code 7 : Stack 뒤집기

```
stack<int> Reverse_Stack( stack<int> X){




}
stack <int> S, T ; // [3, 2, 8, 7, 9, 10, 5]  <--top
for(auto w : {3, 2, 8, 7, 9, 10, 5})
     S.push(w) ;
T = Reverse Stack(S) ; //T는 S의 역순으로 구성된 stack이 되어야 함.,
```

체크 무늬(check pattern) 형식으로 구성된 $N \times N$ "체크" 행렬 M[][]이 있다. 5 by 5 체크 행렬의 예가 아래와 같다. **단 N은 항상 홀수이다.** 아래 예에서 M[1][3]=5이며 M[5][5]=7 이다. 우리는 이런 행렬의 공간을 아끼기 위하여 이 행렬을 1차원 vector[ ] 자료구조에 순서대로, 왼쪽에서 오른쪽으로, 위에서 아래로 저장한다. 아래 행렬에서 빈 공간의 값은 모두 '0(zero)'이다.

	1	*2*	*3*	*4*	*5*
1	3		5		1
2		1		6	
3	7		10		4
4		7		2	
5	2		5		9

M[ ][ ]

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
3	5	1	1	6	7	10	4	7	2	2	5	9

L[ ]

우리는 M[i][j]의 값을 L[]에서 찾으려고 한다. 이 작업을 위한 함수 GetM(int i, int j)를 구성하시오. 단 여러분에게 제공되는 자료구조는 L[ ]이다. 즉 M[i][j] 값을 이를 대체하는 자료구조인 vector L[k]에서 찾아야 한다. 단 L[ ]에는 이미 설명한 대로 자료가 저장되어 있다고 가정한다.

Code 8

```
int N ;            // 행렬의 차원
vector<int> L ;
int GetM(int i, int j) {




} // end of GetM( )
```

Code 9 : 코드의 의미

```
void prob9(){
    vector< vector<int>> X { {3,4,5},{2,3},{6},{10,11,12,13},{22,23}} ;
    int tmp ;

    for(auto it=X.begin()+1; it!= X.end(); it++){
           tmp = (it)->front();
           (it)->erase( (it)->begin() );
           (it-1)->push_back( tmp ) ;
    }
    auto it = X.begin();
    tmp = (it)->front();
    (it)->erase( (it)->begin() );
    (X.end()-1)->push_back( tmp ) ;

} // end of prob7( )
```

위 코드는 이중 vector X에 어떤 작업을 하는 코드이다. 위 작업은 이중 vector에 어떤 작업을 하는 것인지 설명하고, 위 코드가 수행된 뒤에 X의 바뀐 내용을 표시하시오.

X =>

Code 10 : 두 Queue Q1, Q2의 내용을 서로 교환하려고 한다. 즉 그 내용물을 바꾸는 것이다. 단 Q1, Q2는 empty일 경우까지 고려해야 한다. 단 swap( )과 같은 generic algorithm, 또는 추가의 Queue를 만들어서 사용할 수 없으며 아래에서 제공하는 보조 변수, int, float 몇 개만 사용할 수 있다. 즉 아래 코드에서 더 이상 새로운 변수를 추가로 사용할 수는 없다. 아래의 D1, D2는 예를 보여주는 것이므로 이 경우에만 한정해서 해결하는 코드를 작성하면 안된다. 일반적인 크기의 D1, D2에 대하여 적용되도록 작성해야 한다.

```
void prob10(){
    deque<float> D1 {2.3, 5.1, 7.3, 0.5, 9.6, 4.4, 0.7} ;
    deque<float> D2 {15.1, 32.5, -12.3 , 54.6 } ;
    queue<float> Q1(D1), Q2(D2) ; //Q1, Q2에 D1, D2의 내용이 들어있다.
    int size1, size2, idx1, idx2 ;
    float val ;









} // end of prob10( )
```