WEEK 10

Q1.

/ CPP code for bubble sort

// using template function

#include <iostream>

using namespace std;

// A template function to implement bubble sort.

// We can use this for any data type that supports

// comparison operator < and swap works for it.

template <class T> void bubbleSort(T a[], int n)

{

for (int i = 0; i < n - 1; i++)

for (int j = n - 1; i < j; j--)

if (a[j] < a[j - 1])

swap(a[j], a[j - 1]);

}

// Driver Code

int main()

{

int a[5] = { 10, 50, 30, 40, 20 };

int n = sizeof(a) / sizeof(a[0]);

// calls template function

bubbleSort<int>(a, n);

cout << " Sorted array : ";

for (int i = 0; i < n; i++)

cout << a[i] << " ";

cout << endl;

return 0;

}

Output

Sorted array : 10 20 30 40 50

Q4.

/\* C++ Program to find Square function using single template \*/

#include <iostream>

using namespace std;

template <class T>

inline T square(T x)

{

T result;

result = x \* x;

return result;

};

int main()

{

int i, ii;

float x, xx;

double y, yy;

i = 2;

x = 2.2;

y = 2.2;

ii = square<int>(i);

cout << i << ": " << ii << endl;

xx = square<float>(x);

cout << x << ": " << xx << endl;

// Explicit use of template

yy = square<double>(y);

cout << y << ": " << yy << endl;

// Implicit use of template

yy = square(y);

cout << y << ": " << yy << endl;

return 0;

}

***OUTPUT : :***

/\* C++ Program to find Square function using single template \*/

2: 4

2.2: 4.84

2.2: 4.84

2.2: 4.84

Q3.

include

using namespace std;

template

void Swap(T &n1,T &n2)  
{  
T temp;

temp = n1;  
n1 = n2;  
n2 = temp;  
}

int main()  
{  
int i1 = 20, i2 = 40;  
float f1 = 12.5, f2 = 23.7;  
char c1 = ‘c’, c2 = ‘d’;

cout << "Before swapping : i1 ="<< i1 << "\ti2 = "<< i2 << endl;  
cout << "Before swapping : f1 ="<< f1 << "\tf2 = "<< f2 << endl;  
cout << "Before swapping : i1 ="<< c1 << "\ti2 = "<< c2 << endl;

Swap(i1,i2);  
Swap(f1,f2);  
Swap(c1,c2);

cout << "After swapping : i1 ="<< i1 << "\ti2 = "<< i2 << endl;  
cout << "After swapping : f1 ="<< f1 << "\tf2 = "<< f2 << endl;  
cout << "After swapping : i1 ="<< c1 << "\ti2 = "<< c2 << endl;

return 0;

}

output:  
Before swapping : i1 =20 i2 = 40  
Before swapping : f1 =12.5 f2 = 23.7  
Before swapping : i1 =c i2 = d  
After swapping : i1 =40 i2 = 20  
After swapping : f1 =23.7 f2 = 12.5  
After swapping : i1 =d i2 = c

Q5

/\* C++ Program to Find Duplicate Elements in an Array \*/

#include<iostream>

using namespace std;

int main()

{

int i,j,a[50],size;

cout<<"Enter array size( Max:50 ) :: ";

cin>>size;

cout<<"\nEnter array elements :: \n";

for(i=0; i<size; i++)

{

cout<<"\nEnter arr["<<i<<"] Element :: ";

cin>>a[i];

}

cout<<"\nStored Data in Array :: \n\n";

for(i=0;i<size;i++)

{

cout<<" "<<a[i]<<" ";

}

cout<<"\n\nDuplicate Values in Given Array are :: \n\n";

for(i=0; i<size; i++)

{

for(j=i+1;j<size;j++)

{

if(a[i]==a[j])

{

cout<<" "<<a[i]<<" ";

}

}

}

cout<<"\n";

return 0;

}

***OUTPUT : :***

/\* C++ Program to Find Duplicate Elements in an Array \*/

Enter array size( Max:50 ) :: 8

Enter array elements ::

Enter arr[0] Element :: 1

Enter arr[1] Element :: 2

Enter arr[2] Element :: 3

Enter arr[3] Element :: 4

Enter arr[4] Element :: 5

Enter arr[5] Element :: 2

Enter arr[6] Element :: 3

Enter arr[7] Element :: 6

Stored Data in Array ::

1 2 3 4 5 2 3 6

Duplicate Values in Given Array are ::

2 3

Process returned 0

Q9.

|  |
| --- |
| // C++ program to demonstrate the  // function overloading  #include <bits/stdc++.h>  **using** **namespace** std;    // Function to calculate square  **void** square(**int** a)  {      cout << "Square of " << a           << " is " << a \* a           << endl;  }    // Function to calculate square  **void** square(**double** a)  {      cout << "Square of " << a           << " is " << a \* a           << endl;  }    // Driver Code  **int** main()  {      // Function Call for side as      // 9 i.e., integer      square(9);        // Function Call for side as      // 2.25 i.e., double      square(2.25);  **return** 0;  } |

**Output:**

Square of 9 is 81

Square of 2.25 is 5.0625