Experiment 10

Page No.
Date

a) WAP to find sum of Array elements using function template.

→
```
#include <iostream>
using namespace std;
template <typename T>
T arraysum (T& arr[], int n) {
    T sum=0;
  for (int i=0; i<n; i++)
    sum += arr[i];
    return sum;
}
int main() {
  int intArr[] = {1, 2, 3, 4, 5};
  double doubleArr[] = {1.5, 2.5, 3.5};

  cout<<"sum of int array:" << arraysum (intArr, 5) <<endl;
  cout<<"sum of double array:" <<arraysum(doubleArr, 3) <<endl;
}
```

Output:

```
sum of int array: 15
sum of double array: 7.5
```

b. WAP of square function using template specialization.
- calculate square of integer no. & a string.
(square of string is nothing but concatenation of a string with itself.

- Write a specialized function for the square of a string.

→
```
#include <iostream>
using namespace std;
template <class T>
T square (T n) {
    return n*n;
}
template < >
string square (string s) {
    return s+s;
}
int main() {
    int n;
    string str;
    cout << " enter an integer: ";
    cin >> n;
    cout << " square of integer: " << square (n) << endl;

    cout << " enter a string: ";
    cin >> str;
    cout << " square of strings: " << square (str) << endl;

    return 0;
}
```

Page No.
Date

. output :

```
Enter an integer :  2
Square of integer :  4
Enter a string :  abc
square of string:  abcabc.
```

c WAP to build simple calculator using class template.
(: using 10 operations & switch case in main())

->

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

template <class T>
class calc {
  T a, b;

Public:
  calc (Tx, Ty) {

      a = x;
      b = y;
   }
void add() {
   cout<<" sum = " << a+b <<endl;
  }

Void subtract() {

   cout <<" difference = " << a-b <<endl;
  }
```

```
void mul() {
    cout << " Product = " << a*b << endl;
}

void div() {
    cout << " Quotient = " << a/b << endl;
}

void mod() {
    cout << " Remainder = " << (int) a % (int) b << endl;
}

void sinV() {
    cout << " sin(a) = " << sin(a) << ", sin(b) = " << sin(b) << endl;
}

void cosV() {
    cout << " cos(a) = " << cos(a) << ", cos(b) = " << cos(b) << endl;
}

void tanV() {
    cout << " tan(a) = " << tan(a) << ", tan(b) = " << tan(b) << endl;
}

void logV() {
    cout << " log(a) = " << log(a) << ", log(b) = " << log(b) << endl;
}

void root() {
    cout << " root(a) = " << sqrt(a) << ", root(b) = " << sqrt(b)
    << endl;
}
```

```
};

int main () {
    int ch ;
    double x, y;
    cout << " enter two numbers: ";
    cin >> x >> y;
    calc <double > c (x,y);

cout << " \n1. add \n 2. subtract \n 3. multiply \n 4.
        divide \n5. modulus \n6. sin \n 7. cos \n8. Tan
        \n 9. log \n10. Root \n Enter choice: ";

    cin>>ch;

switch (ch) {

 case 1 :
      c.add ();
      break;

 case 2 :
      c.subtract();
      break;

 case 3:
      c.mul();
      break;

 case 4 :
     c.div ();
     break;
```

```
case 5:
    c.modC();
    break;

case 6:
    c.sinvC();
    break;

case 7:
    c.cosvC();
    break;

case 8:
    c.tanvC();
    break;

case 9:
    c.logvC();
    break;

case 10:
    c.rootC();
    break;

default:
    cout<<" Invalid choice:";
}
    return 0;
}.
```

Page No.
Date

. Output :

Enter two numbers : 2 4

1. Add
2. substract
3. Multiply
4. Divide
5. Modulus
6. sin
7. cos
8. Tan
9. Log
10. Root

Enter choice : 2
Difference : -2.

d WAP to implement Push & Pop from stack using class template.

→

```
#include <iostream>
using namespace std;
template <class T>
class stack {
    Ts[10];
    int top;
Public:
    stack() {
      top = -1;
    }
```

```
void push (T x) {
    if (top == 9)
    cout << "overflow\n";
    else s[++top] = x;
    }

void pop() {
    if (top == -1)
    cout << "underflow\n";
    else
     cout << "popped: " << s[top--] << endl;
    }

void display() {
    for(int i=top; i>=0; i--)
    cout << s[i] << " ";
    cout << endl;
    }
};
int main() {
    stack<int> st;
    int ch, x;
    do {
     cout << "\n1.Push 2.Pop 3.display 4.exit \n Enter choice:";
     cin >> ch;
     switch (ch) {
        case 1: cout << "enter value:";
                cin >> x;
                st.push(x);
                break;
```

```
case 2:
    st.pop();
    break;

case 3:
    st.display();
    break;
}
}
while (ch!=4);
}
```

- Output :

1. Push 2. Pop 3. display 4. exit

enter choice : 1
enter choice : 10

1. Push 2. Pop 3. display 4. exit.

enter choice : 1
enter choice : 20.

1. Push 2. Pop 3. display 4. exit

enter choice : 3
enter choice : 20 10

Page No.
Date

Experiment 11

* WAP to implement generic vectors. Include following member functions.

- To create the vector
  a) to modify the value of a given element.
  b) to multiply by scalar value.
  c) to display the vector in form (10, 20, 30).

→

```
#include <iostream>
using namespace std;
template <class T>          // class is generic
class vector {
    T a[10];                // array to store up to 10 elements
    int n;
Public:
  void create() {           // create function
    cout<<"enter size:";
    cin>>n;
    cout<<" enter elements:";
    for(int i=0; i<n; i++)
    cin>>a[i];
  }
  void modifyElement (int pos, T val) {   // to change the values
      if (pos>=0 && pos<n)                // position
        a[pos] = val;
        else
          cout<<" invalid position\n";
  }
  void multiplyscalar (T s) {
  for(int i=0; i<n; i++)
    a[i] *= s;
}
```

eg (10, 25, 30) X 2
⇒ (20, 50, 60)

```
void display() {
    cout << "(";
    for (int i=0; i<n; i++) {
        cout<<a[i];
        if (i<n-1)
        cout << ", ";
    }
cout << ")\n";
}
};
int main() {
  vector <int> v; - creates a vector of integers
  int ch, pos, val, s;
  v.create();
  do {
    cout <<"\n1.modify element 2.multiply by scalar 3.
          display 4. Exit \n Enter choice:";
    cin >> ch;
    switch (ch) {
      case 1:
        cout <<" enter position and new value:";
        cin>>pos>>val;
        v.modifyElement(pos,val);
        break;

      case 2:
        cout <<" enter scalar value:";
        cin>>s;
        v.multiplyscalar(s);
        break;

      case 3: v.display();
```

```
        break;
    }
    while (ch != 4);
}
```

- Output:

```
Enter size: 3
enter elements: 10 20 30

1. Modify element  2. Multiply by scalar  3. display  4. exit
  enter choice: 3
    (10, 20, 30)
```

b With Iterators :

```
#include <iostream>
#include <vector>
using namespace std;
template <class T>
class MyVector {
    vector <T> v = {10, 20, 30};
  public:
    void modify (int i, T val)
    {
        v[i] = val;
    }
    void multiply (T s) {
      for (typename vector <T> :: iterator it = v.begin ();
            it != v.end ();
            ++ it)
          *it = (*it) * s;
```

```
}
void display() {
    cout << "(";
    for (typename vector <T>:: iterator it = v.begin(); it != v.end();
        ++it) {
            cout << *it;
            if (it != vend() - 1) ;
            cout << ", ";
            }
            cout << ")\n";
        }
    };
int main() {
    MyVector <int> v;
    v.display();
    v.modify(1, 50);
    v.display();
    v.multiply(2);
    v.display();
}
```

output :

(10, 20, 30)

(10, 50, 30)

(20, 100, 60)

Page No.
Date

Experiment - 12

* WAP using STL

a) Implement Stack
b) Implement Queue
c) Implement sorting & searching with user-defined records such as Person Record (Name, birth date, telephone no), item Record (item name, code, Quantity & cost)

a)

```
#include <iostream>
#include <stack>
using namespace std;
int main () {
stack <int> s;
s.push(10);
s.push(20);
s.push (30);
cout<<" stack elements (top to bottom):";
while( !s.empty ()) {
        cout<<s.top() <<" ";
        s.pop();
}
}
```

- Output:

Stack elements (top to bottom): 30 20 10

b)

```
#include <iostream>
#include <queue>
using namespace std;
int main () {
    queue <int> q;
    q.Push (10);
    q.Push (20);
    q.Push (30);
cout << "Queue elements (front to rear): ";
while (!q.empty()) {
    cout << q.front () << " ";
    q.pop();
  }
}
```

- Output:

Queue elements (front to rear): 10 20 30

c)

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;
struct person {
  string name;
  string birthDate;
  string phone;
};
bool compareByName (const person &a, const person &b) {
    return a.name < b.name;
}
int main () {
```

```
vector<Person> people = {{"Alice", "01-01-2000", "111"},
{"bob", "02-02-1999", "222"}, {"charlie", "03-03-2001", "333"}};
sort(people.begin(), people.end(), compareByName);
cout << "sorted list by name:\n";
for (auto &p : people)
    cout << p.name << " " << p.birthDate << " " << p.phone
    << "\n";
    string searchName = "Bob";
    auto it = find_if (people.begin(), people.end(), [&](
    person &p) { return p.name == searchName;
    }
    if (it != people.end())
    cout << "\nfound: " << it->name << " " << it->birthDate << " " << it->
    phone << "\n";
else
    cout << "\nperson not found\n";
}
```

- Output:

```
sorted list by name:
Alice 01-01-2000 111
Bob 02-02-1999 222
charlie 03-03-2001 333

Found: Bob 02-02-1999 222
```