19K-0281.

OOP FINAL

Date: _____

NAME: ANUSHA SAAD

SECTION: H

ID : 19K-0281.

QUESTION NO 1:-

1. a) error is that just one general type T is made in function while in main two different type are passed int and float.

b)
```
template<typename T1, typename T2>
void Print_max(const T1 &a, const T2 &b){
}
```

2.

a)
```
Class A{
    int x;
    Public:
        display(){
        cout<<x<<endl; cout<<getvalue()<<endl;
    }
        void setvalue(int x){
            this->x = x;
    }
        int getvalue(){
            return x;
    }
int main
```

Date: _____

```
void set(){
    A a;
    a.setvalue(10);
    a.display();

}
```

3)

a)

x is a private variable. so cant access it directly in class B.

b)

```
class A{
    int x;
    Public:
        display(){
            cout<<x;
        }
    int getvalue(){
            return x;
        }
};
class B{
    A a;
    Public:
        display(){
            cout<<a.getvalue()<<endl;
        }
};
```

Date: _____

4) Class Shape {
int x;
Public:
Shape (int c, int d, int e): Circle(d,e)
{
x = c;
}
};
Class Circle : virtual public Shape {
int y;
Public:
Circle (int g, int w) : square(w)
{ y=g };
};
Class square : virtual public shape {
int z;
Public: square(int j) {
z = j; } };

Class circle_on_square : Public circle, public square{
int a;
Public:
circle_on_square (int l, int m, int n, int o):
Shape(m, n, o){
a = l; }
};
int main(){
circle_on_square cc(1, 2, 3, 4);
}

Date:

QUESTION NO 2:-

a) the major critisim on friend Keyword is if a class is made a friend of other class it can access all of its private protected variables and functions. it finishes the concept of encapsulation.

b)
```
class A{
    Public:-
        void display(){
        cout<<" A "<<endl } };
class B: Public A{
    Public:
void display(){
    cout<<" B "<<endl}
};
```

this will print //A from class B.

b) int main(){
A* ob = new B;
ob.display();

this will display A from class A ~~ev~~. although B. for this purpose a keyword virtual is used.

virtual void display(){
- - - - - -
}.

c) because The proper syntax to make a class abstract or pure virtual function is so

```
Class Temp {
    Public:
        Temp(){ };
        virtual void abc() = 0;
};
```

~~The virtual function in given que~~ the return type is missing.
in the case given in the paper it was a normal virtual function call.
The virtual func is given question won't be compile due to the missing definition.

Date:

d) A child class always have access to its base ~~pu~~ class public and protected members. Public members of a class can be accessed by an class or function - that is allowed.

```
class A{
    int x;
    Public:
    void display(){
        cout<<getvalue()<<endl;}
    int getvalue(){
        return x;
        }
};
Class B: Public A{
    Public:  A ob;
        void display(){
            ob.display();}
};
```

e) We cannot overload the function on return type because the compiler wont be able to determine which method to executed at runtime.

```
int main()
    A(6, 8);
}
int A(int a, int y){
    return x; }
```

```
int A(int x, int y)
{ return y;}
```

the compiler will not know which to call because of same parameter

Date: _____

f)

because when we open a file using ifstream functions, it open in txt mode by default.
∴ Program runs fine.

the system decide the mode of file through its extension -

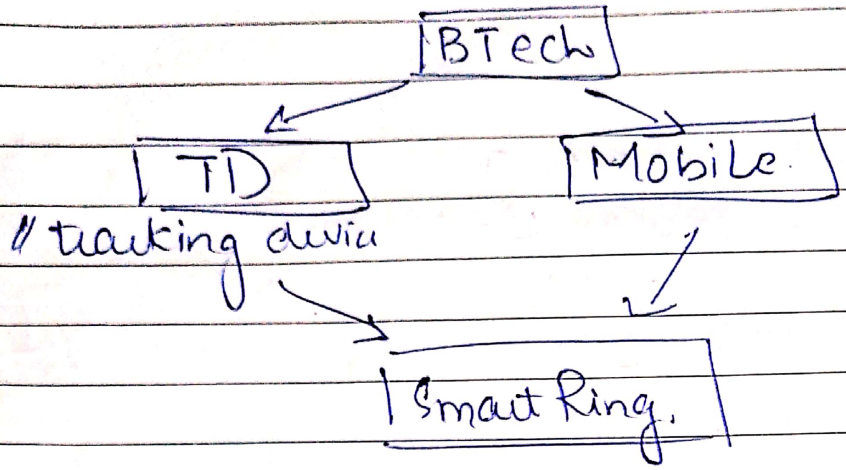
QUESTION NO 4:-

(1)

```
Class BTech
    Protected:
        String model;
        Int yeaw;
    Public:
};

class LED : Public BTech {
    int no of_supp_app;
    Public:
};

class Mobile : Public BTech {
    double camera;
    Public:
};

Class Tablet : Public BTech {
    double screen_size;
    Public:
};

Class TD : Public BTech {
    float accuray;
};

Class SmartRing : Public Mobile, Public
                                TD {
};
```

Date: _____

(2) Yes, diamond problem has occurred and it can be resolved using hybrid inheritance



(3) global func.:

```
void item_sort() {
    SmartRing S[10];
    RedTech R[10];
    for(int i=0; i<10; i++) {
    if(S[i].year() > S[i+1].year()) {
        S[i].year = S[i+1].year() } }

    for(int j=0; j<10; j++){
      if (R[i].year > R[i+1].year) {
          R[i].year = R[i+1].year ;
    }
```

(4) class virtual TD : Public BTech {
// other code
friend class RedTech;
};

class Red Tech {

// Red Tech being friend of tracking device (TD) can access all private member of TD //.
};

QUESTION NO # 4. 19K-0281.

page # 12.

5)

```
Class RedTech {
                //      4
Public:
    void checker(TD  &ob.) {
try { if (obj.year  < 2010){
                throw (exception);
catch(...)      }
                cout << "exception caught" << endl;
};
```

QUESTION NO 5 ~~Ans~~.

19K-0281

QUESTION NO 3:-

```
(1) class employer{
(2)     int u_ID;
        string type;
        Public:
        virtual void post_vacancy = 0.

};
  class candidate {
     string, name, NIC, deg, Dob, exp, add;
     float salary, u_ID;
     static_ int Tot;

     Public:
        friend class Moderator;
        candidate (){
            tot+ =tot; }
     };
int Candidate :: tot = 0;
```

```
Class Ed: Public employer {
    int camp , teachy;
    bool c;
    Public:
    Ed(){};
      void post_vacancy(){
        cout << "teaching years and ability
        to cope with pressure" << endl;
      }
};
Class Pharmacy: Public Eemployer
{
  float bud;
  String skill;

  Public:
      void post_vacancy(){
      cout << "good analytical skills" << endl;
    }
};
Class Bank: Public employer{
    int bran;
    String skill;
    bool working;
    Public:
  void Post_vacancy(){
      cout << "communication skills and the
      candidate can work in large
      Teams" << endl;};
};
```

Date:

```
Class construction: Public Employer {
    int Proj;
    String ability;

    Public:
        void post_vacancy() {
        cout << " Ability to work in remote
                areas " << endl;}
};

Class Moderator {
    int mod_id;
    Public:
(4)  void operator<(candidate c1, candidate c2){
    if (c1.getexpe() < c2.getexpe())
     { cout << " 2nd cand is more exp "; }
    else{
        cout << " 1st cand is more exp ";
(5)  } cout << " total no of candidata "
                << candidate :: tot << endl;
    };
```

3)(employer class.)

```
Class employer{
    Public:
        employer (int v_id){
            char ch;
            fstream in;
            in.open("myfile.Txt", ios::in |ios:out)
```

```
while (!in.eof()) {
    if(in){
        ch = in.get()
        cout << ch << endl;
        ch++;
        in<<ch;
    }
}

in.close();
cout << " uniqueID = " << u_id << endl;
cout << ch << endl;
this -> u_id = u_id;
}
```

D)

```
class employer {

public:
    void Recieve_App (candidate can) {
        c.select_cand((can));
        cout << " canaidate selected " << endl;
```