94

Name:_

PART I - 10 SHORT QUESTIONS (20 Points: 2 points for each Question)

For the following ten questions circle the letter "T" or the letter "F" to indicate whether or not each of the statements is true or false, respectively.

1. T (F) : An exception that is not caught is ignored. Thus the program continues its execution normally.

2. (T) F : catch(...) is used to catch any exception type.

3. (T) F : throw with no arguments rethrows the same exception type.

4. (T) F : The Seekg() and seekp() functions are used to reposition the *"file position pointer"*.

5. T (F) : The insertion operator "<<" can be used with both sequential and random-access files.

6. (T) F : Data can be modified in a random access file without destroying other data in the file.

7. T (F) : A stack is first-in, first-out data structure

8. T (F) : A derived class pointer can point to an object of the base class without the need for casting.

9. (T) F : Derived class can provide their own implementation of a base class virtual function. But if they do not, the base class implementation is used.

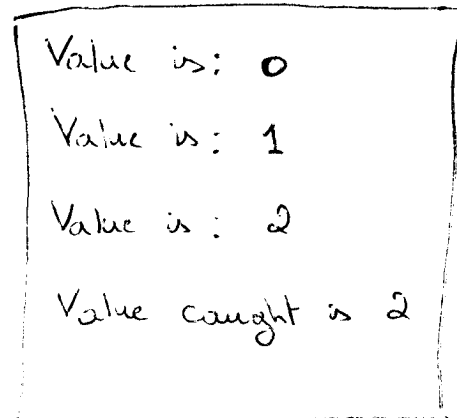10. (T) F : Any destructor in a class hierarchy can be declared virtual.

PART II - 7 PROBLEMS (35 Points)

1. What is the output of the following program when it is executed? (5 Points)

```cpp
#include <iostream.h>
void main() {
  try {
    for(int val=0; val<6; val++) {
      cout << "Value is: " << val << endl;
      if ( val > 1 )
        throw(val);
    }
  }
  catch(int v) {
    cout << "Value caught is: " << v << endl;
  }
  catch(...) {
    cout << "Value 0, 1, 2 are caught\n";
  }
}

void catch_Zero(int v) {
  cout << "One is caught\n";
}

void catch_Two(int v) {
  cout << "Two is caught\n";
}
```

Value is: 0

Value is: 1

Value is: 2

Value caught is 2

2. What is the output of the following program when it is executed? (5 Points)

```cpp
#include <iostream.h>
void bar(int j) {
  if (j < 0)
    throw(j);
  cout << "Result is: " << 100 / j << endl;
}
void foo(int i) {
  bar(i);
  cout << "Value is:" << i << endl;
}
```

5

```
void main() {
  foo(10);

  try {
      foo(-1);
  }
  catch(char * s) {
      cout << "Devide by zero\n";
  }
  catch(int v) {
      cout << "No can do!\n";
  }
}
```

Result is: 10

Value is: 10

No can do!

3. **What is the output of the following program when it is executed? (5 Points)**

```
#include <iostream.h>
#include <fstream.h>
struct finalS {
    int vOne;
    double vTwo;
};
void main() {
  finalS fs;

  ofstream ofObj("final.dat", ios::binary);

  for (int i=1; i <=10; i++) {
      fs.vOne = i;
      fs.vTwo = i + 0.5;
      ofObj.write(reinterpret_cast<const char*>(&fs),sizeof(finalS));
  }
  ofObj.close();

  ifstream ifObj("final.dat", ios::ate|ios::binary);

  ifObj.seekg(sizeof(finalS),ios::beg);
  ifObj.read(reinterpret_cast<char *>( &fs),sizeof(finalS));
  cout << fs.vOne <<endl << fs.vTwo << endl;

  ifObj.read(reinterpret_cast<char *>( &fs),sizeof(finalS));
  cout << fs.vOne <<endl << fs.vTwo << endl;

}
```

9
9.5
10
10.5

9

4. **Given a data file "TEST.DAT" with 6 records of different integer values**
   What is printed following the execution if this program? (5 Points)

test.dat
8 7
4 3
1 0
2 3
6 5
1 2

```cpp
#include <iostream.h>
#include <fstream.h>

class myFile {
 public:
   myFile(char * nm) {
      ifo.open(nm,ios::in);
   }
   void mk() { bm = ifo.tellg();}
   void ps(long mk) { ifo.seekg(mk,ios::beg);}
   void go() { ps(bm);}
   int gt() {
      int i;
      ifo >> i;
      return i;
   }
 private:
   ifstream ifo;
   long bm;
};

void main() {
   myFile mf("test.dat");

   cout << mf.gt() << " - ";
   cout << mf.gt() << " - ";
   cout << mf.gt() << " - ";
   mf.mk();
   mf.ps(0);
   cout << mf.gt() << " - ";
   mf.go();
   cout << mf.gt() << endl;
}
```

57 - 43 - 10 - 87

87 - 43 - 10 - 87 - 23

5. **What is the output of the following program when it is executed?** (5 Points)

```cpp
#include <iostream.h>
template <class T>

class arrC {
 public:
   arrC(int sz ) {
      szV = sz;
      arr = new T[sz];
   }
   void fl(T);
   void ot() const;
 private:
   int szV;
   T inV;
   T *arr;
};
```

```
template <class T>
void arrC<T>::fl(T in) {
  inV = in;
  for (int i=0; i < szV; i++)
    arr[i] = i + inV;
}

template <class T>
void arrC<T>::ot() const {
  for (int i=0; i < szV; i++)
    cout << arr[i] << " ";
}

void main() {
  arrC<double> dArr(5);
  arrC<int> iArr(6);

  dArr.fl(0.7);
  dArr.ot();
  cout << endl;

  iArr.fl(90);
  iArr.ot();
}
```

0.7 1.7 2.7 3.7 4.7

90   91   92   93   94   95

6. **What is the output of the following program when it is executed? (5 Points)**

```
#include <iostream.h>
class anml {
 private:
   int wt;
   double ht;
 public:
   anml(int w=0, double h=0.0) {
     wt = w >0 ? w : 0;
     ht = h > 0.0 ? h : 0.0;
   }
   virtual void a2b() {
     cout << "Run or Fly from A to B\n";
   }
   virtual void sng() {
     cout << "I cannot\n";
   }
   virtual void inf(){
     cout << wt << " " << ht << endl;
   }
};
```

```
class dg : public anml {
 public:
   dg(int a, int w, double h) : anml(w,h) {
      ag = a;
   }
   void a2b() {
      cout << "I Run from A to B\n";
   }
   void inf(){
      anml::inf();
      cout << ag << endl;
   }
 private:
   int ag;
};

class brd : public anml {
 public:
   brd(bool f, int w, double h) : anml(w,h) {
      fl = f;
   }
   void a2b() {
      if (fl > 0)
         cout << "I walk from A to B\n";
      else
         cout << "I Fly from A to B\n";
   }
   void sng() {
      cout << "Skweek\n";
   }
 private:
   int fl;
};

void main() {
  brd tt(1, 5, 10.4);
  dg mx(7, 20, 20.3);
  anml *aP, &aR=mx;

  mx.sng();
  tt.a2b();
  aP = &tt;
  aP->inf();
  aP->a2b();
  aP->sng();

  aR.inf();
  aR.a2b();
  aR.sng();
}
```

I cannot

I walk from A to B

5   10.4

I walk from A to B

Skweek

20   20.3

7

I Run from A to B

I cannot

001 / 6 - 12

7. **What is the output of the following program when it is executed? (5 Points)**

```cpp
#include <iostream.h>

class dnC {
public:
  dnC(int i=0) { st(i);}
  void st(int i) {val=i;}
  int gt() const {return val;}

private:
  int val;
};

class abC {
 public:
  abC(int nb=2) {
    sz = nb;
    comP = new dnC[nb];
  }
  void prt(int x=10) const {
    if (x < sz)
      cout << comP[x].gt() << " ";
    else
      cout << "Noting to print...";
  }
  void fl(int i=10) {
    for (int j=sz; j >=0; j--)
      comP[j].st(j+i);
  }
 private:
  int sz;
  dnC *comP;
};

void main() {
  abC obj(6);

  obj.prt(1);
  obj.fl(100);
  obj.prt(6);
  obj.prt(3);
  obj.prt(0);
}
```

PART III - THREE PROBLEMS (10 + 17 + 18 Points)

1. Write a program to read an integer number from a keyboard, and then to search for it in a sequential data file called "value.dat" that contains records of one integer field.

   The program should also write to the screen the word **'FOUND'** or the word **'NOT-FOUND'** depending on the results of the search.

Note: Use the space bellow to write your answer. You may use the back of the page if necessary.

```
# include < iostream.h>
# include < fstream.h>
# include < stdlib.h>


void main ()
{
    ifstream n ("value.dat", ios::in);

    n.seekg (0, ios::beg);

    int a;

    while ( n >> a )


Void main ()
{
    cout <<" Enter an integer: ";

    int x;

    cin >> x;

    ifstream n ("value.dat", ios :: in);

    n.seekg (0, ios::beg),

    int a;

    if (!n)
    {
        cout << "Could not open file \n";
```

10

2. Having a random data file named " salary.dat" that contains information about a company's employees and their monthly salaries. The file records have the following structure: Employee Id (an integer.),
          Employee first name (maximum of 20 characters.),
          Employee family name (maximum of 20 characters.),
          Employee monthly salary (a double).

The company has decided to increase the monthly salary of all employees by a percentage of **5.3%**.

Write a C++ program that will *adjust the monthly salary information **in the "salary.dat" file*** to meet the company's salary increase decision.

Implement all the necessary error check to earn full grades.

Note: Use the space bellow to write your answer. You may use the back of the page if necessary.

```cpp
# include <iostream.h>

* include <fstream.h>

* include <stdlib.h>

struct employee
{
    int id;
    char first[20];
    char last[20];
    double salary;
}

void main ()
{

ifstream n ("salary.dat", ios::in);
if (!n)
    {
    cout << "Could not open file \n";
    exit (1);
    }

n. seekg (0, ios::beg);

employee blank;

n. read (reinterpret_cast < char *> (& blank), sizeof (emplayee));
```

3. Given the following geometric categories, with these definitions:

Car
Truck
Vehicle
Sedan (a kind of sports car)
Wagon (a kind of carriage for example to sell on it vegetables)

Knowing that all object have names, maximum speed and number of passengers (the driver is included), moreover they have two member function print the name and move:
a) Write the deepest possible inheritance hierarchy
b) Write only the following member functions:
  ▪ **Constructors** for each class
  ▪ **printName** to print the name of the form
  ▪ **move** to go forward ot backward or left or right in meters
  ▪ Overloaded **operator <<** to print the data members value of the object

c) Declare **static objects** in main and call the appropriate functions
d) Declare **dynamic objects** in main and call the appropriate functions to prove the inheritance as well as polymorphism.

Note: Use the space bellow to write your answer. You may use the back of the page if necessary.

```
* include <iostream.h>

* include <string.h>

* include <iomanip.h>


class vehicle
{ friend ostream & operator << (ostream & , vehicle);
  protected:

      char name[30];
      int mspeed;

      int nps;

  public:

      vehicle (char *a , int b , int c)
      {
        strcpy (name ,a);

        mspeed = b;

        nps = c;

      {

  virtual void printName () const = 0;
```

```cpp
ostream & operator << ( ostream & out, truck z )
{
    out << static_cast <vehicle> (z);
    return out;
}


class car : public vehicle
{
    friend ostream & operator << ( ostream & , car );
    public:
        car ( char * a, int b, int c ): vehicle (a, b, c)
        {}
        virtual void print() const { cout << " car " << endl; }
        virtual void move() const { cout << "forward" << endl; }
};

ostream & operator << ( ostream & out, car z )
{
    out << static_cast <vehicle> (z);
    return out;
}


class sedon : public car
{
    friend ostream & operator <<( ostream & , sedon );
    public:
        sedon ( char * a, int b, int c )
```

```cpp
void move() const { cout << "Speedy" << endl; }

};


ostream & operator << ( ostream & out, sedan z)
{
    out << static_cast <car> (z);

    return out;

}



Void main ()
{
    car    m("Fiat", 180, 5);
    truck   n("Toyota", 160, 3);
    Sedan   o("corvette", 300, 2);
    wagon   s("Ferrari", 20, 1);

    m. print();

    n. move();

    o. print();

    s. move();

    vehicle * ptr = 0;

    ptr = & m;
```

```
ptr = & O;

ptr → move ();


}
```