Problem Solving using c and c++

Lab Assignment 6

Patil Amit Gurusidhappa
19104004
B11

Lab Assignment: 6 (Exception Handling)

**1. Exceptions are run-time errors or anomalies that programs encounter at some point in its
execution. To handle exceptions, an exception handling mechanism is used. Exception handling is used so that the normal flow of the programs can be maintained even after runtime
errors. Write C++ programs to implement exception handling mechanism using-
(i) Multiple catch statements**

```
#include <iostream.h>
#include <conio.h>

void test(int x)
{
    try
    {
        if (x > 0)
            throw x;
        else
            throw 'x';
    }
    catch (int x)
    {
        cout << "Catch a integer and that integer is:" << x;
    }
    catch (char x)
    {
        cout << "Catch a character and that character is:" << x;
    }
}

void main()
{
    clrscr();
```

```
    cout << "Testing multiple catches\n:";
    test(10);
    test(0);
    getch();
}
```

(ii) Re-throwing an exception
(iii) By specifying exceptions
(iv) By invoking a function that generates exception

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

double division(int a, int b)
{
    if (b == 0)
    {
        throw "Division by zero condition!";
    }
    return (a / b);
}

int main()
{
    int x = 50;
    int y = 0;
    double z = 0;

    try
    {
        z = division(x, y);
        cout << z << endl;
    }
    catch (const char *msg)
    {
        cerr << msg << endl;
    }

    return 0;
```

```
}
```

2. User defined exceptions are defined by inheriting and overriding exception class. Write a C++ program to implement the concept of user defined exceptions.

```
#include <iostream>
#include <exception>
using namespace std;
class MyException : public exception
{
public:
    const char *what() const throw()
    {
        return "Attempted to divide by zero!\n";
    }
};
int main()
{
    try
    {
        int x, y;
        cout << "Enter the two numbers : \n";
        cin >> x >> y;
        if (y == 0)
        {
            MyException z;
            throw z;
        }
        else
        {
            cout << "x / y = " << x / y << endl;
        }
    }
    catch (exception &e)
    {
        cout << e.what();
    }
}
```

3. Identify and correct the error(s) in the following code fragment and also write its output.

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

class A
{
public:
    A()
    {
    }
    class B
    {
    };
    class C : public virtual B
    {
    };
    class D : public virtual B
    {
    };
    void getData()
    {
        throw D();
    }
};
int main()
{

    A a1;
    try
    {
        a1.getData();
    }
    catch (A ::B &)
    {
        cout << "In class A";
    }
    catch (A ::C &)
```

```
    {
        cout << "In class B";
    }
    catch (A ::C &)
    {
        cout << "In class C";
    }
    catch (...)
    {
        cout << "None";
    }
}
```

```
        catch (A ::C &)
        ^~~~~
In class A
PS E:\Work\JIIT\sem_6\JIIT-SEM-6\
Problem_solving_lab\Lab6>
```

4. What will be the output of the following code?

(i)

```
#include <iostream>
#include <exception>
using namespace std;
int main()
{
    int x = 10, y = 20, z = 30;
    float d;
    try
    {
        if ((x - y) != 0)
        {
            d = z / (x - y);
            cout << d;
        }
        else
        {
```

```
            throw(x - y);
        }
    }
    catch (int m)
    {
        cout << "Infinite " << m;
    }
}
```

```
                            > cd "e:\
Work\JIIT\sem_6\JIIT-SEM-6\Proble
m_solving_lab\Lab6\" ; if ($?) {
g++ q4_1.cpp -o q4_1 } ; if ($?)
{ .\q4_1 }
-3
PS E:\Work\JIIT\sem_6\JIIT-SEM-6\
Problem_solving_lab\Lab6>
```

(ii)

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

class B
{
};
class D : public B
{
};
int main()
{
    D d;
    try
    {
        throw d;
    }
    catch (B b)
```

```
    {
        cout << "Base Exception";
    }
    catch (Derived d)
    {
        cout << "Derived Exception";
    }
    return 0;
}
```

(iii)

```
#include <iostream>
#include <exception>
using namespace std;
class T
{
    static int c;
    int i;

public:
    T()
    {
        c++;
        i = c;
        cout << "Creating object " << i;
        if (i == 4)
            throw 4;
    }
    ~T() { cout << "Deleting object " << i; }
};
int T::c = 0;
int main()
{
    try
    {
        T a[5];
    }
    catch (int id)
```

```
    {
        cout << "Caught " << id;
    }
}
```

```
Work\JIIT\sem_6\JIIT-SEM-6\Proble
m_solving_lab\Lab6\" ; if ($?) {
g++ q4_3.cpp -o q4_3 } ; if ($?)
{ .\q4_3 }
Creating object 1Creating object
2Creating object 3Creating object
 4Deleting object 3Deleting objec
t 2Deleting object 1Caught 4
PS E:\Work\JIIT\sem_6\JIIT-SEM-6\
Problem_solving_lab\Lab6>
```