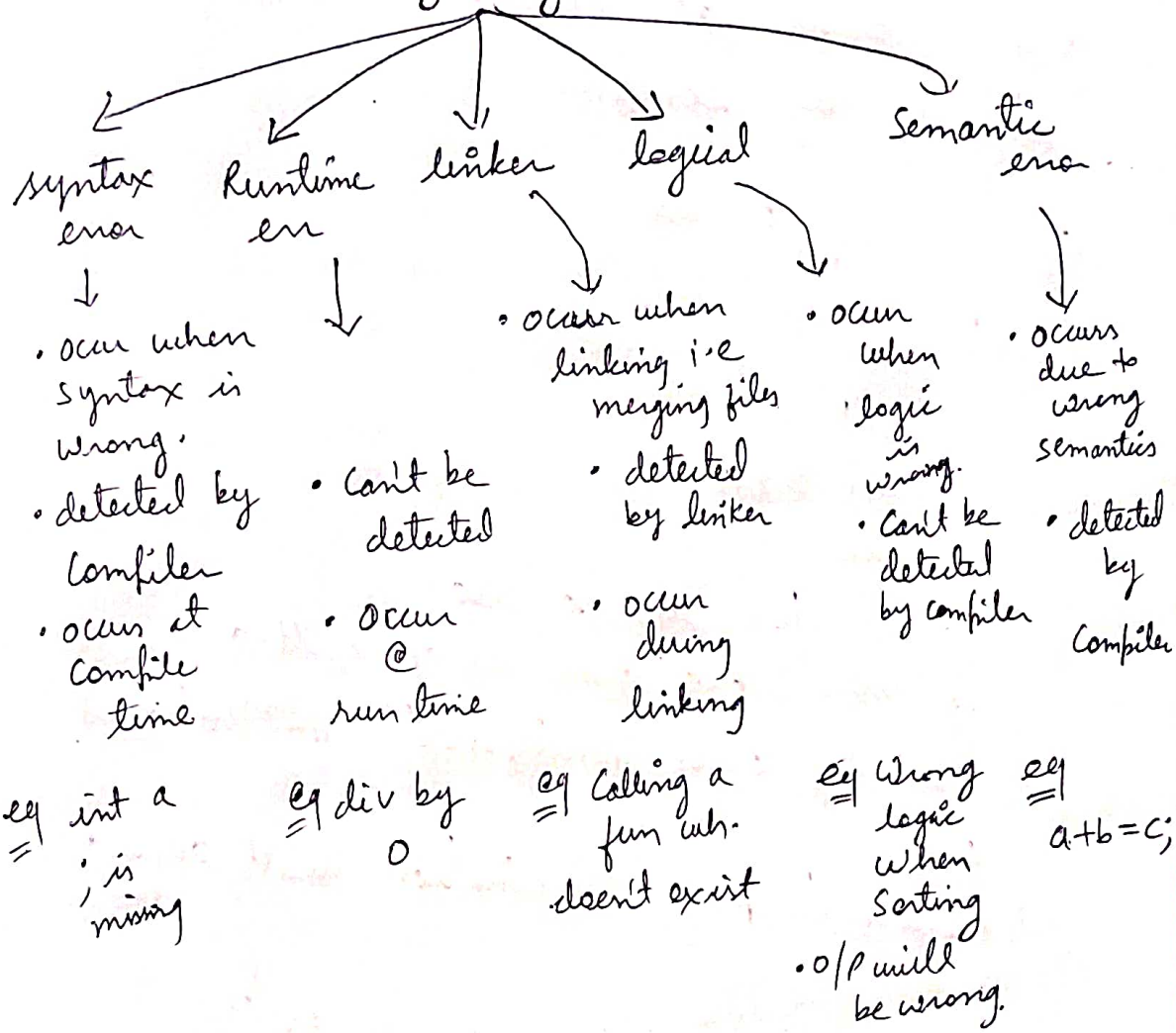
Exception Handling

(Unit-5)

Types of errors



Q) What r exceptions? (Ex)

A. Ex r "unpredictable runtime err"

- They r unpredictable as you can't predict by running whether ex will occur or not.

eg Div by zero.

- Only when user i/p the number 0 ex will occur. So you can't predict whether user will enter 0 or not.
- They occur at runtime. So can't be detected by compile.

eg Div by 0 ex.

```
int main()
{
    int a;
    cin >> a;
    cout << 5/a;
// 5 is div by a no.
// if user i/p 0, then excep will occur
// Div by 0 is "undefined", mathematical ex.
// Abnormal prog. termination will occur.
```

Q) what r some other eg. of ex?

A) 1) Reading a file wch doesn't exist
2) Opening a link wch is invalid
3) sq. root of -ve No.
4) div by 0
5) m/m not available during dy. m/m alloc.
6) declaring an array of undefined size in C is an ex.

eg int arr[n]; → value of n is not known at compile time.

Q) what happens when an Ex occurs?

A) · Abnormal prog termination will occur.
· If the ex is not handled APT will occur.
· To avoid this, ex must be handled.

Q) how to handle the ex?

A) There r two ways to handle an ex.
1) using simple if, else
2) using a "try catch" block

Note *
· In both cases, there r two requirements must be fullfilled to handle the ex.
1) you must know in wh. part of the code ex can occur.
2) u must know the condition wh. will result in ex.

eg In case of "div by 0",
· The condition at wh. ex occurs is

```
if(a==0)
```

· The code in wh. ex occurs is

```
cout<<5/a;
```

Using if else to handle the exception

- This prog shows how to handle div by 0 using if, else.

```
int main()
{
    int a;
    cin >>a;        //Cond" to
    if(a==0)        //check for ex
    {
        cout<<" Cant div by 0"; } //handle ex
        return 0;
    }
    else cout << a/5; //run this code.
}
```

explanation :-

- Check for cond" in wh. ex can occur using an if.
- run the ~~statmen~~ statement under else.
- using this simple if, else ex can be handled.
- If user enters, APT will not occur, but "div by 0" err msg will be displayed.

Using try & catch block to handle ex

- C++ provides a feature called try & catch block to handle ex.
- The stmt of try & catch is as follows

```
try
{
    <condn where ex may occur>
        throw <val>;

    <code>

}
catch (a ___ )
{
    // err msg or err handling code.

}
```

- Try block contains the condition in wch ex may occur.
- It will have a throw statement wch is executed when any ex occurs.
- A value is specified to the throw stmt.
- when throw stmt is executed, the prog will move to the catch block & this val will be passed to the catch handler.

- Catch block will handle the ex.
- Catch block will take one param as an array argument.
- The value p/v to the throw ___ will be assigned to the catch param.
- The value will represent err code.
- Any value can be specified to a throw stmt..

eg handle div by 0 using try & catch

```
int main()
{
    int a;
    cin>>a;
    try
    {
        if(a==0) //cond" in wh. err occurs
            throw 100; // throw this value
                       // it can have any
                       // value.
        cout<<5/a;
        //if no ex is thrown, then this
        code will run
    }
    catch (int x)
    {
        Cout<<"Can't div by 0";
        Cout<<"err code "<<x;
    }
}
```

throw stmt will jump here

err handler.

Note . try & catch will not detect any error auto^{y} & it will not handle er by itself

. You must specify the cond^{n} at wh. er will occur inside try block

. You must handle er appropriately inside catch.

Q) Why is try & catch block used if it doesn't detect er & handles er by itself

A) Uses of try & catch

1) It separates er handling code from normal prog code.

. Catch block (handler) is separate & it will handle exc.

2) Fun can handle any ex they choose.

. Fun can be restricted to throw certain excep.

3) Grouping ~~error~~ of error types

. classes can be combined w/ ex. to create user defined ~~ex types~~ error.

. ex can be categorized by using hierarchy of classes / ex. obj.

example: Handle an ex such occurs when file can't be opened.

A)
```
int main()
{
    //create an i/p stream
    ifstream myin("file.txt");

    try
    {
        if (!myin) //file can't be opend.
            throw 100; //throw ex with value 100.

        Char words[255]
        myin.read(words,          //continue if no ex
            sizeof(words));

        Cout << words;
        myin.clr()
    }
    catch (int x)
    {
        Coult << "cant open file"
        Cout << "er code" << x;
    }
}
```

//Err handler when file can't be opened.

explanation :-

- After creating an obj & opening the file the stream state will be checked inside try block.
- If false, then file can't be opened. So ex will be thrown by "throw 100".
- ex will be handled by the catch block.
- catch block will print ex msg.

Throwing Mul ex

- In some cases, mul ex needs to be thrown
- So there must be mul catch handler for ~~there~~ each ex.
- To throw mul ex., the data type of each ex shud be different

eg throw 100; // diff data type for
throw 'a'; diff ex.

For eg:- suppose "div by 0" as well as "div by 1" is considered as an ex.

Prog to handle these 2 ex.

```
int main()
{
    int a;
    cin >> a;
    try
    {
        if (a == 0)
            throw 100;        } when a = 0, throw an int value the exact value doesn't matter
        if (a == 1)
            throw 'a';        } when a = 1 throw a char value. the exact value doesn't matter
        cout << a/5;
    }
    catch (int x)
    {
        cout << "div by 0";
        cout << "err msg" << x;     } //handler-1 for int ex
    }
    catch ( ~~int~~ char x)
    {
        cout << "div by 1";
        cout << "err code" << x;     } //handler 2 for char ex.
    }
```

Explanation :-

1) inside try there r 2 condition
   if (a == 0) & if (a == 1)
2) throw value should be of diff data type for each ex.
3) there will be 2 catch blocks / handlers. They will catch / handle ex of diff data type.

Note : Throw value doesn't matter, the data type of throw value matters.

- It means if 2 diff ex throw ~~same~~ diff value of same data type eg both int (100, 200), then they will be handled by same handler
- If two ex. throw values of diff data types, then they will be handled by diff handlers.
  eg (100, 'a');