# CS19001: Programming and Data Structures Laboratory

Aritra Hazra;
CSE, IIT Kharagpur

`http://cse.iitkgp.ac.in/~aritrah/course/lab/PDS/Autumn2019/`

16-Aug-2019

# The If-else syntax

```
if (boolean combination of relational expressions)
{ ⟵ scope of if
   program statement(s)
}
else
{ ⟵ scope of else
   program statement(s)
}
```

# Nested If-else

```
if(condition1)
{
  if(condition2)
    {
      statement(s);
    }
  else
    { /* execute when
      (condition1 && !condition2)==TRUE */
      statement(s);
    }
}
else
{
  statement(s);
}
```

# Compute $|xy|$

```
if (((x>=0)&&(y>=0))||((x<0)&&(y<0)))
     z = x * y;
else
     z = -x * y;
```

- Complex conditions (error pron and non-intuitive in some cases)

# Compute $|xy|$

```
if (x>=0)
{
     if (y>=0) z = x*y; else z = -(x*y);
}
else
{
     if (y>=0) z = -(x*y); else z = x*y;
}
```

- This is how we think, create sub cases based on simple conditions

# Repeated If-else

```
if(condition1){
  statement(s);  /* go to college */
}
else{
  if(condition2){
    statement(s); /* otherwise watch movie*/
  }
  else{
    if(condition3){
      statement(s); /* otherwise, sleep*/
    }
    else{
      ------/*thinking with elimination*/
    }
  }
} /*nesting of conditions is only in else*/
```

# Repeated If-else has a simpler syntax

```
if(Condition 1){
  Block 1
}
else if(Condition 2){
  /*else if = same as saying otherwise */
  Block 2
}
------------
------------
else if(Condition n){
  Block n
}
else{
  Block n+1
}
/*need not manage complex hierarchy of braces*/
```

# Implementation of the assignment $y = |x|$

```
scanf("%d",&x);
if (x == 0)
  y = 0;
else if (x > 0)
  y = x;
else y = -x;
```

# Multiway ($>$ 2) Program flow

- Still now, a program had atmost two possible execution paths at any point of time
- We simply tested boolean conditions
- Similarly, we can try matching an expression with possible set of values it may assume
- The set of possible values have to be known while writing the program

# The switch statement

```
switch (E) {
   case val1 :
     Block 1  /*Execute if E = val1*/
     break;
   case val2 :
     Block 2
     break;
   ........
   case valn :
     Block n
     break;
   default    :
   /* Execute if E is equal to none */
     Block n+1
}
```

# The break in switch

- In a switch case, once a match is found, further comparisons are disabled.
- But all following statements before the closing brace are executed one by one.
- The break in switch forces exit from switch

# Thank You