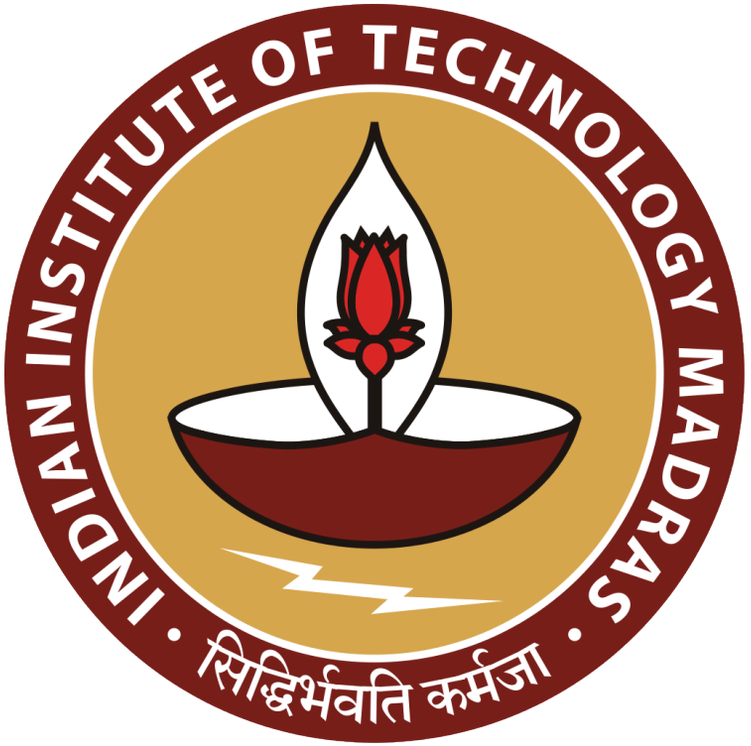


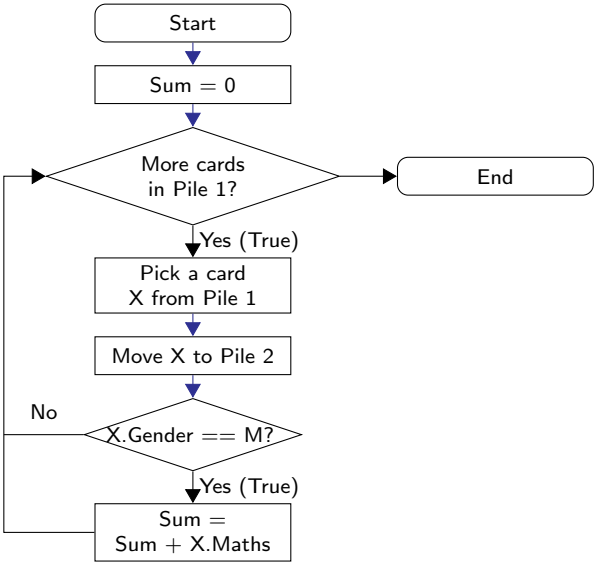
IIT Madras

ONLINE DEGREE

Pseudocode: Procedures

Sum of Boys' Maths marks

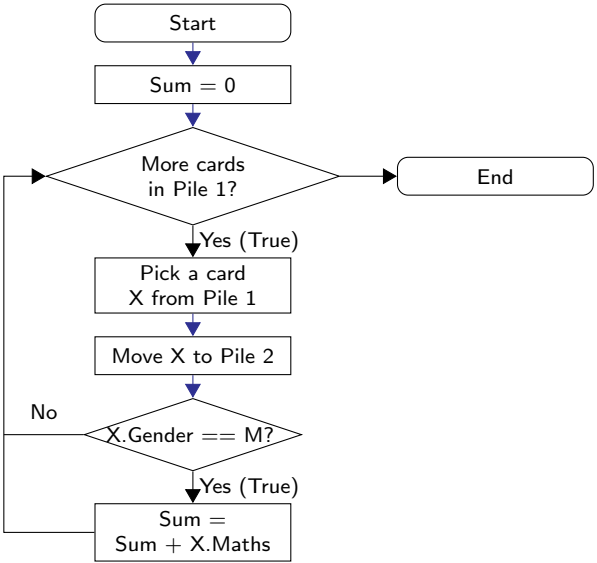
```
Sum = 0
while (Pile 1 has more cards) {
  Pick a card X from Pile 1
  Move X to Pile 2
  if (X.Gender == M) {
    Sum = Sum + X.Maths
  }
}
```



Sum of Boys' Maths marks

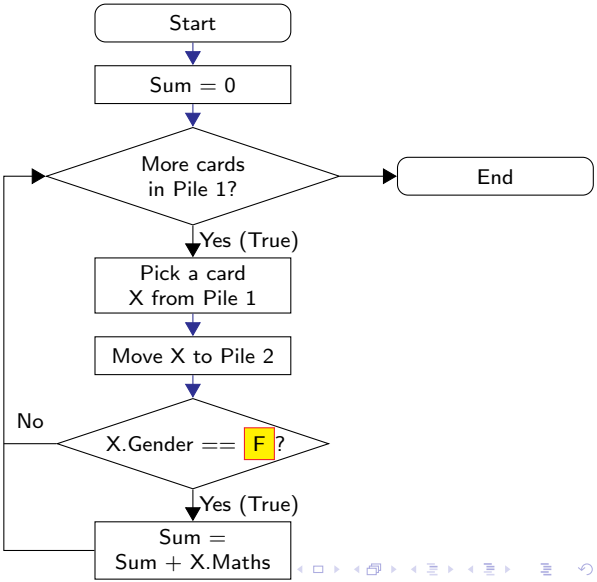
```
Sum = 0
while (Pile 1 has more cards) {
  Pick a card X from Pile 1
  Move X to Pile 2
  if (X.Gender == M) {
    Sum = Sum + X.Maths
  }
}
```

- What if we want to sum Maths marks of girls?



Sum of Girls' Maths marks

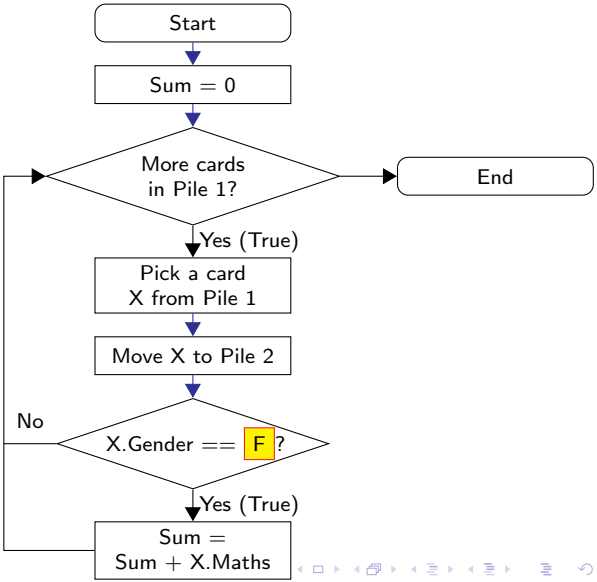
```
Sum = 0
while (Pile 1 has more cards) {
  Pick a card X from Pile 1
  Move X to Pile 2
  if (X.Gender == F) {
    Sum = Sum + X.Maths
  }
}
```



Sum of Girls' Maths marks

```
Sum = 0
while (Pile 1 has more cards) {
  Pick a card X from Pile 1
  Move X to Pile 2
  if (X.Gender == F) {
    Sum = Sum + X.Maths
  }
}
```

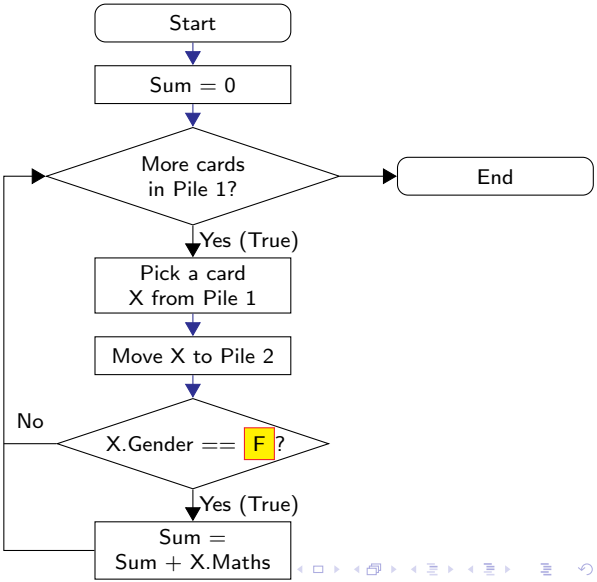
- Only change is the value we check for **X**.Gender



Sum of Girls' Maths marks

```
Sum = 0
while (Pile 1 has more cards) {
  Pick a card X from Pile 1
  Move X to Pile 2
  if (X.Gender == F) {
    Sum = Sum + X.Maths
  }
}
```

- Only change is the value we check for **X**.Gender
- Remaining pseudocode is identical



A procedure to sum up Maths marks

```
Procedure SumMaths(gen)
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen) {
        Sum = Sum + X.Maths
      }
    }
    return(Sum)
end SumMaths
```

A procedure to sum up Maths marks

- Procedure name: **SumMaths**

Procedure SumMaths(gen)

```
Sum = 0
while (Pile 1 has more cards) {
  Pick a card X from Pile 1
  Move X to Pile 2
  if (X.Gender == gen) {
    Sum = Sum + X.Maths
  }
}
return(Sum)
```

end SumMaths

A procedure to sum up Maths marks

- Procedure name: **SumMaths**
- Argument receives value: **gen**

```
Procedure SumMaths( gen )
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen ) {
        Sum = Sum + X.Maths
      }
    }
    return(Sum)
end SumMaths
```

A procedure to sum up Maths marks

- Procedure name: **SumMaths**
- Argument receives value: **gen**
- Call procedure with a parameter **SumMaths(F)**

```
Procedure SumMaths(gen)
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen) {
        Sum = Sum + X.Maths
      }
    }
    return(Sum)
end SumMaths
```

A procedure to sum up Maths marks

- Procedure name: **SumMaths**
- Argument receives value: **gen**
- Call procedure with a parameter **SumMaths(F)**
- Argument variable is assigned parameter value

```
Procedure SumMaths(gen)
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen) {
        Sum = Sum + X.Maths
      }
    }
    return(Sum)
end SumMaths
```

A procedure to sum up Maths marks

- Procedure name: **SumMaths**
- Argument receives value: **gen**
- Call procedure with a parameter **SumMaths(F)**
- Argument variable is assigned parameter value
- Procedure call **SumMaths(F)**, implicitly starts with

gen = F

```
Procedure SumMaths(gen)
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen) {
        Sum = Sum + X.Maths
      }
    }
    return(Sum)
end SumMaths
```

A procedure to sum up Maths marks

- Procedure name: **SumMaths**
- Argument receives value: **gen**
- Call procedure with a parameter **SumMaths(F)**
- Argument variable is assigned parameter value
- Procedure call **SumMaths(F)**, implicitly starts with **gen = F**
- Procedure returns the value stored in **Sum**

Procedure SumMaths(gen)

```
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen) {
        Sum = Sum + X.Maths
      }
    }
    return(Sum)
```

end SumMaths

A procedure to sum up Physics marks

```
Procedure SumPhysics(gen)
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen) {
        Sum = Sum + X.Physics
      }
    }
    return(Sum)
end SumPhysics
```

A procedure to sum up Physics marks

- Only change is the field we examine in the card

  X.Physics, instead of **X**.Maths

Procedure SumPhysics(gen)

```
Sum = 0
while (Pile 1 has more cards) {
  Pick a card X from Pile 1
  Move X to Pile 2
  if (X.Gender == gen) {
    Sum = Sum + X.Physics
  }
}
return(Sum)
```

end SumPhysics

A procedure to sum up Physics marks

- Only change is the field we examine in the card

  X.Physics, instead of **X**.Maths
- For Chemistry, add up **X**.Chemistry
- For Total, add up **X**.Total

Procedure SumPhysics(gen)

```
Sum = 0
while (Pile 1 has more cards) {
  Pick a card X from Pile 1
  Move X to Pile 2
  if (X.Gender == gen) {
    Sum = Sum + X.Physics
  }
}
return(Sum)
```

end SumPhysics

A procedure to sum up Physics marks

- Only change is the field we examine in the card

  X.Physics, instead of **X**.Maths
- For Chemistry, add up **X**.Chemistry
- For Total, add up **X**.Total
- Pass field name as parameter

Procedure SumPhysics(gen)

```
Sum = 0
while (Pile 1 has more cards) {
  Pick a card X from Pile 1
  Move X to Pile 2
  if (X.Gender == gen) {
    Sum = Sum + X.Physics
  }
}
return(Sum)
```

end SumPhysics

A procedure to sum up any type of marks

```
Procedure SumMarks(gen,fld)
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen) {
        Sum = Sum + X.fld
      }
    }
    return(Sum)
end SumMarks
```

A procedure to sum up any type of marks

- Two parameters, gender (**gen**) and field (**fld**)

Procedure SumMarks(gen, fld)

```
Sum = 0
while (Pile 1 has more cards) {
  Pick a card X from Pile 1
  Move X to Pile 2
  if (X.Gender == gen) {
    Sum = Sum + X.fld
  }
}
return(Sum)
```

end SumMarks

A procedure to sum up any type of marks

- Two parameters, gender (**gen**) and field (**fld**)
- **gen** is assigned a value, M or F, to check against **X**.gender

```
Procedure SumMarks(gen,fld)
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen) {
        Sum = Sum + X.fld
      }
    }
    return(Sum)
end SumMarks
```

A procedure to sum up any type of marks

- Two parameters, gender (**gen**) and field (**fld**)
- **gen** is assigned a value, M or F, to check against **X**.gender
- **fld** is assigned a field name, to extract appropriate card entry **X**.fld

```
Procedure SumMarks(gen,fld)
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen) {
        Sum = Sum + X.fld
      }
    }
    return(Sum)
end SumMarks
```

A procedure to sum up any type of marks

- Two parameters, gender (**gen**) and field (**fld**)
- **gen** is assigned a value, M or F, to check against **X**.gender
- **fld** is assigned a field name, to extract appropriate card entry **X**.fld
- Single procedure **SumMarks** to handle different requirements

```
Procedure SumMarks(gen,fld)
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen) {
        Sum = Sum + X.fld
      }
    }
    return(Sum)
end SumMarks
```

A procedure to sum up any type of marks

- Two parameters, gender (**gen**) and field (**fld**)
- **gen** is assigned a value, M or F, to check against **X**.gender
- **fld** is assigned a field name, to extract appropriate card entry **X**.fld
- Single procedure **SumMarks** to handle different requirements
  - **SumMarks(F,Chemistry)**
    Sum of Girls' Chemistry marks
  - **SumMarks(M,Physics)**
    Sum of Boys' Physics marks
  - ...

```
Procedure SumMarks(gen,fld)
    Sum = 0
    while (Pile 1 has more cards) {
      Pick a card X from Pile 1
      Move X to Pile 2
      if (X.Gender == gen) {
        Sum = Sum + X.fld
      }
    }
    return(Sum)
end SumMarks
```

Calling a procedure

- Use procedure name like a math function, as part of an expression

GirlChemSum = SumMarks(F,Chemistry)

BoyChemSum = SumMarks(M,Chemistry)

if (GirlChemSum $\geq$ BoyChemSum) {

  "Congratulate the girls"

}

else {

  "Congratulate the boys"

}

Calling a procedure

- Use procedure name like a math function, as part of an expression
- Assign the return value to a variable

```
GirlChemSum = SumMarks(F,Chemistry)
BoyChemSum = SumMarks(M,Chemistry)
if (GirlChemSum ≥ BoyChemSum) {
    "Congratulate the girls"
}
else {
    "Congratulate the boys"
}
```

Calling a procedure

- Use procedure name like a math function, as part of an expression
- Assign the return value to a variable
- A procedure may not return a value

GirlChemSum = SumMarks(F,Chemistry)

BoyChemSum = SumMarks(M,Chemistry)

if (GirlChemSum $\geq$ BoyChemSum) {

  "Congratulate the girls"

}

else {

  "Congratulate the boys"

}

Calling a procedure

- Use procedure name like a math function, as part of an expression
- Assign the return value to a variable
- A procedure may not return a value
- Correct marks for one subject on a card
  - **Procedure UpdateMarks(CardId, Subject, Marks)**

GirlChemSum = SumMarks(F,Chemistry)

BoyChemSum = SumMarks(M,Chemistry)

if (GirlChemSum $\geq$ BoyChemSum) {

  "Congratulate the girls"

}

else {

  "Congratulate the boys"

}

Calling a procedure

- Use procedure name like a math function, as part of an expression
- Assign the return value to a variable
- A procedure may not return a value
- Correct marks for one subject on a card
  - **Procedure UpdateMarks(CardId, Subject, Marks)**
- Procedure call is a separate statement

```
GirlChemSum = SumMarks(F,Chemistry)
BoyChemSum = SumMarks(M,Chemistry)
if (GirlChemSum ≥ BoyChemSum) {
   "Congratulate the girls"
}
else {
   "Congratulate the boys"
}
```

```
Sum = 0
...
UpdateMarks(17,Physics,88)
...
GirlChemSum =SumMarks(F,Chemistry)
```

Summary

- Procedures are pseudocode templates that work in different situations

Summary

- Procedures are pseudocode templates that work in different situations
- Delegate work by calling a procedure with appropriate parameters

Summary

- Procedures are pseudocode templates that work in different situations
- Delegate work by calling a procedure with appropriate parameters
  - Parameter can be a value, or a field name

Summary

- Procedures are pseudocode templates that work in different situations
- Delegate work by calling a procedure with appropriate parameters
  - Parameter can be a value, or a field name
  - SumMarks(M,Total)

Summary

- Procedures are pseudocode templates that work in different situations
- Delegate work by calling a procedure with appropriate parameters
  - Parameter can be a value, or a field name
  - SumMarks(M,Total)
- Calling a procedure

Summary

- Procedures are pseudocode templates that work in different situations
- Delegate work by calling a procedure with appropriate parameters
  - Parameter can be a value, or a field name
  - SumMarks(M,Total)
- Calling a procedure
  - Procedure call is an expression, assign return value to a variable

Summary

- Procedures are pseudocode templates that work in different situations
- Delegate work by calling a procedure with appropriate parameters
  - Parameter can be a value, or a field name
  - SumMarks(M,Total)
- Calling a procedure
  - Procedure call is an expression, assign return value to a variable
    - GirlsChemSum = SumMarks(F,Chemistry)

Summary

- Procedures are pseudocode templates that work in different situations
- Delegate work by calling a procedure with appropriate parameters
  - Parameter can be a value, or a field name
  - SumMarks(M,Total)
- Calling a procedure
  - Procedure call is an expression, assign return value to a variable
    - GirlsChemSum = SumMarks(F,Chemistry)
  - No useful return value, procedure call is a separate statement

Summary

- Procedures are pseudocode templates that work in different situations
- Delegate work by calling a procedure with appropriate parameters
  - Parameter can be a value, or a field name
  - SumMarks(M,Total)
- Calling a procedure
  - Procedure call is an expression, assign return value to a variable
    - GirlsChemSum = SumMarks(F,Chemistry)
  - No useful return value, procedure call is a separate statement
    - UpdateMarks(17,Physics,88)

Summary

- Procedures are pseudocode templates that work in different situations
- Delegate work by calling a procedure with appropriate parameters
  - Parameter can be a value, or a field name
  - SumMarks(M,Total)
- Calling a procedure
  - Procedure call is an expression, assign return value to a variable
    - GirlsChemSum = SumMarks(F,Chemistry)
  - No useful return value, procedure call is a separate statement
    - UpdateMarks(17,Physics,88)
- Procedures help to modularize pseudocode

Summary

- Procedures are pseudocode templates that work in different situations
- Delegate work by calling a procedure with appropriate parameters
  - Parameter can be a value, or a field name
  - SumMarks(M,Total)
- Calling a procedure
  - Procedure call is an expression, assign return value to a variable
    - GirlsChemSum = SumMarks(F,Chemistry)
  - No useful return value, procedure call is a separate statement
    - UpdateMarks(17,Physics,88)
- Procedures help to modularize pseudocode
  - Avoid descibing the same process repeatedly

Summary

- Procedures are pseudocode templates that work in different situations
- Delegate work by calling a procedure with appropriate parameters
  - Parameter can be a value, or a field name
  - SumMarks(M,Total)
- Calling a procedure
  - Procedure call is an expression, assign return value to a variable
    - GirlsChemSum = SumMarks(F,Chemistry)
  - No useful return value, procedure call is a separate statement
    - UpdateMarks(17,Physics,88)
- Procedures help to modularize pseudocode
  - Avoid descibing the same process repeatedly
  - If we improve the code in a procedure, benefit automatically applies to all procedure calls