Program Development and Problem Solving

Program Development and Problem Solving

- Program
  - sequence of instructions that must be followed to solve a particular problem
- Computer programming
  - enables us to use the computer as a problem solving tool

Stages of Program Development

1. Problem analysis and specification
2. Data organization and algorithm design
3. Program coding
4. Execution and testing
5. Program maintenance

Stages of Program Development

- Today:
  1. Problem analysis and specification
  2. Data organization and algorithm design
- When we start programming:
  3. Program coding
  4. Execution and testing
  5. Program maintenance

Problem Analysis and Specification

- Specification
  - description of the problem's input
    - what information is given and which items are important in solving the problem
  - description of the problem's output
    - what information must be produced to solve the problem
- Analysis
  - generalize specification to solve given problem + related problems of same kind

Data Organization and Algorithm Design

- Data organization
  - representation of the input and output
  - variables
    - assignment of names to various quantities, which may assume different values
- Algorithm design
  - development of procedures or *algorithms* to process the input and produce the required output

Algorithm Design and Refinement

- Basic description of an algorithm
  - get input values
  - compute output values for the given input
  - return output values
- Algorithm refinement
  - adding problem specific details
  - for example, computation of output values
  - a formula or equation may be required

Example – Adding two numbers

Problem: Calculate 13+23

- Generalization of problem
  - Calculate the sum of two numbers

Adding two numbers

- Specification

Inputs
- 13
- 23

Output
- sum of 13 and 23

- Generalization

Inputs
- first number
- second number

Output
- sum of first number and second number

Adding two numbers

- Data organization: assign variable names to inputs and outputs

num1	• variable name to represent the first number
num2	• variable name to represent the second number
sum	• variable name to represent sum of first number and second number

Adding two numbers

- Basic algorithm
  - get values for *num1* and *num2*
  - calculate *sum* for the given *num1* and *num2*
  - return the value of *sum*
- Refinement
  - get values for *num1* and *num2*
  - calculate *sum = num1 +num2*
  - return the value of *sum*

Algorithm: Adding two numbers

Algorithm to compute the sum of two numbers

Input: *num1*, *num2*
Output: *sum*

1. Get *num1,num2*
2. Calculate *sum = num1 +num2*
3. Return *sum*

Control Structures

- Determine the sequence of execution of the program's instructions
  - Sequential execution: Steps are performed in order, each step being executed once
  - Selective execution: One of a number of alternative actions is selected and executed
  - Repetitive execution: One or more steps are performed repeatedly

Minimum of two numbers

Problem: Find the minimum of two numbers

- Selective execution
  - either the first number is less than the second number or the second number is less than the first

Minimum of two numbers

- Inputs
  - *num1* • first number
  - *num2* • second number
- Output
  - *min* • minimum of first, second numbers
- How do we find the minimum??
  - if the first number is smaller than the second number, then the first number is the minimum
  - else, the second number is the minimum

Minimum of two numbers

- We express the computation more formally as follows:

```
if (num1 < num2)
    min = num1
else
    min = num2
```

Algorithm: Minimum of two numbers

Algorithm to compute the minimum of two numbers

Input: *num1*, *num2*
Output: *min*

1. Get *num1*,*num2*
2. Calculate *min*:
   if (*num1* < *num2*)
   min = *num1*
   else
   min = *num2*
3. Return *min*

Sum between two integers

Problem: Compute the inclusive sum between two integers

- Examples
  - the inclusive sum between 2 and 6 is 2+3+4+5+6 = 20
- Repetitive execution
  - start at first integer, add next integer to cumulative sum until second integer is reached

Sum between two integers

- Inputs
  - *int1* • first integer
  - *int2* • second integer
- Output
  - *sum_between* • inclusive sum between first integer and second integer
- Additional variables
  - *k* • counter

Sum between two integers

- We express the computation more formally as follows:

  sum_between = 0
  for *k = int1* to *int2*
  sum_between = sum_between + k

- We need to assign an initial (starting) value for *sum_between*

Algorithm: Sum between two integers

Algorithm to compute the sum between two integers

Input: *int1, int2*
Output: *sum_between*
Variables: *k*

1. Get *int1, int2*
2. Calculate *sum_between*:
   sum_between = 0
   for *k = int1* to *int2*
   sum_between = sum_between + k
3. Return *sum_between*

In-class problem

- Design a generalized algorithm to solve the problem

Calculate the average exam score for a class of 10 students