**Compiler Project Options**

1. To obtain a minimal passing grade on the project:

Implement a compiler for a simple language which allows

assignments, all types of arithmetic expressions, input and

output. Flag the first error, and quit. There must be I/O and

comments.

2. To obtain a C-, C or C+

Include if-then-else statements and loops. Including

arrays will give you a high C grade. The better your

error correction the higher the grade.

3. To obtain B-, B or B+

Include functions and procedures. You do not need to

support recursion. Including arrays will give you a higher grade.

Include error correction. Without error correction and

arrays you will get a B- if other parts are working.

4. To get A or A-

Implement the whole project.

program Option1;

!here is a program which does nothing

var first, second, third : integer;

var x: integer;

Begin

write('enter first ');

read(first); !who's on first

second := 12; !what's on second

third := first + second - 23; !I dunno' s on third

third := third + 2\*second + 1;

x := first / 4;

write(first);

write('\n');

write(second);

write('\n');

write(third);

write('\n');

write(x);

write('\n')

end.

Program Average;

var sum, average,count, number : integer;

Begin

sum := 0;

count := 0;

write('Enter numbers-- end with 999');

read(number);

while( number <> 999) do

begin

count := count + 1;

sum := sum + number;

read(number)

end;

if count = 0 then

write('cannot divide by zero')

else

begin

average := sum/count;

write('the average (whole number) is');

write(average)

end

end.

Program printarray;

var size, data, count : integer;

var x: array[10]; !assumes array is indexed from 1

begin

Write('Enter the size');

read(size);

Write('Enter data');

count := 0;

while (count < size ) do

begin

count := count + 1;

read(data);

x[count] := data

end;

count:= size;

Write('In reverse \n');

while count >= 1 do

begin

write(x[count]);

write('\n');

count := count - 1

end

end.

program Option3;

var amt:integer;

procedure addemup(amount:integer); !averages "amount" numbers

var

sum, average, count, number: integer;

Begin ! procedure

sum := 0;

count := 0;

write('Enter numbers');

while (count <= amount) do

begin

read(number);

count := count + 1;

sum := sum + number

end;

if count = 0 then

write('cannot divide by zero')

else

begin

average := sum/count;

write('the average (whole number) is');

write(average)

end

end; ! procedure addemup

Begin

write('How many numbers would you like to read');

read (amt);

addemup(amt)

end.

program FeetAndInches;

var input, whichone : integer;

procedure FeetToInches(feet:integer);

var inches : integer;

begin

inches := 12\* feet;

Write('The number of inches is ');

write(inches)

end;

procedure InchesToFeet(inches:integer);

var

feet, remainder: integer;

begin

Write('The number of feet is ');

feet := inches/12;

Write(feet);

write(' with a remainder of ');

remainder := inches - 12\*feet;

write(remainder);

write(' inches ')

end;

begin

write('enter 1 for feet to inches any number otherwise ');

read(whichone);

if whichone = 1 then

begin

write('Enter number of feet ');

read(feet);

feettoinches(feet)

end

else

begin

write('Enter number of inches ');

read(inches);

inchestofeet(inches)

end

end.

program bubblesort;

var values: array[10];

var size,x,temp: integer;

procedure bubblesort(size:integer);

var tempswap, i,k:integer;

begin

i:=0;

k:=0;

write('In Sort\n');

while i<(size) do

begin

k:=0;

while k<(size-1) do

begin

if values[k]>values[k+1] then

begin

tempswap:=values[k+1];

values[k+1]:=values[k];

values[k]:=tempswap

end;

k:=k+1

end;

i:=i+1

end

end;

Begin

x:=0;

write('How many values do you want to enter(Max 10)?');

read(size);

if size > 10 then

write('Max size is 10\n')

else

begin

while x < size do

begin

write('Enter number: ');

read(temp);

values[x]:=temp;

x:=x+1

end;

bubblesort(size);

x:=0;

write('Sorted:\n');

while x < size do

begin

write(values[x]);

write('\n');

x:=x+1

end

end

end.

program recursive; ! adds up the first n positive numbers

var number, answer: integer;

function add(n:integer): integer;

var add : integer;

begin

if n = 1 then

add := 1

else

add := n + add(n-1)

end;

begin

write('Enter a positive number');

read(number);

answer := add(number);

write(answer)

end.

program hanoi ;

var

height: integer;

procedure move(start, goal, extra, height : integer);

begin

if height > 0 then

begin

move(start, extra, goal, height-1);

write(start);

write(' to ');

write(goal);

write('\n');

move(extra,goal,start,height-1)

end

end;

begin

write('How many disks / ');

read(height);

move(1,3,2,height)

end.