Exercise 2: Expresions,Variables,Assignments

PRAVEEN KUMAR R

5/2/18

Assignment	1
Reg No	312217104114
Name	PRAVEEN KUMAR R
Grade	
Date	19-02-2018

1 Objective

1. Translate expressions to C.
2. Declare variables of data types appropriate for the calculation.
3. Order the update of variables using a sequence of assignments.
4. Use alternative and conditional statements.
5. Specify, define, and call simple functions.

2 Home work

3 Area and Perimeter of circle

Problem Description

Write a program to calculate the area and the perimeter of a circle. Read the radius from the user and print the outputs on the display.

Specification

`area(r)` input: r of tynope float output: a value of type float calculated as pi*r*r. `perimeter(r)` input: r of type float output: a value of type float calculated as 2* pi*r

Program Design

no The program consists of the following.

1. `area(r)`: to compute the area of the circle with radius r.

2. `perimeter(r)`: to compute the perimeter of the circle with radius r.
3. `main()`:reads the radius and store it in r, calls the functions `area(r)`,`perimeter(r)` and prints the result.

Program

```
#include<stdio.h>
/*
  input: r of type float
  output: a value of type float calculated as pi*r*r.

*/
float area(float r)
{
  return 3.14*r*r;
}
/*
  input: r of type float
  output: a value of type float calculated as 2* pi*r
*/
float perimeter(float r)
{
  return 2*3.14*r;
}
int main()
{
  float r,ar,p;
  scanf("%f",&r);
  ar=area(r);
  p=perimeter(r);
  printf("radius=%4.2f\t area=%4.2f\t perimeter=%4.2f\t",r,ar,p);
  return 0;
}
```

```
radius=40.00    area=5024.00    perimeter=251.20
```

4 Leap Year

Program Description

Write a Boolean function `is_leap()` for testing whether a year is leap year or not. Test the function from `main()`.

Specification

input: year, an variable of type int output: FLASE, if the given year is not a leap year TRUE, if the given year is a leap year

Program Design

The program consists of the following.

1. `is_leap(year)`: returns `true` if the year is leap and returns `false` if the year is not leap.
2. `main()`: inputs a year, tests the function `is_leap(year)` and prints 1 if the year is leap and prints 0 if the year is not leap.

Algorithm

is_{leap}(int year) if (year%4==0 and year%100!=0 or year%400==0) return true else return false

Program

```
#include<stdio.h>
#include<stdbool.h>
bool is_leap(int year)
{
  if (year%4==0 && year%100!=0 ||year%400==0)
    return true;
  else
    return false;
}
int main()
{
  int year=1999,leap;
  // scanf("%d",&year);
  leap=is_leap(year);
  printf("%s",leap?"leap year":"not leap year");
}
```

test input

```
1999
```

output

```
not leap year
```

5 Roots of Quadratic equation

Program Description

Read the coefficients `a`, `b`, and `c` of a quadratic equation. Calculate the discriminant. Define a function `sign()` that returns -1 or 0 or 1 for a negative number, zero or a positive number, respectively. Use it to test the discriminant. If the discriminant is non-negative, find the roots of the equation, and print them. Avoid duplicate calculations whereever possible.

Specification

Function `sign(n)` takes the parameter n in float and returns one of the integers -1,0 or 1.
`int sign(float n)`

Program Design

The program consists of the following.

1. `sign(n)`:returns 1 if n is positive, 0 if n is 0 and -1 if n is negative.
2. `main()`:inputs a, b and c of a quadratic equation, then finds the determinant `det`, tests the function `sign(n)` by passing det as argument. If `sign(n)` returns 1 or 0, finds the roots and prints them. If it retuns -1, prints **imaginary roots**.

Program

```
#include<stdio.h>
#include<math.h>
int sign(float n)
{
  if (n>0)
    return 1;
  else if(n==0)
    return 0;
  else
    return -1;
}
int main()
{
  int a,b,c,m;
  double det,r1,r2;
  // scanf("%d%d%d",&a,&b,&c);
  a=3;b=-6;c=-9;
  det=(b*b)-(4*a*c);
  m=sign(det);
  if (m==1)
    {
      r1=(-b+sqrt(det))/(2*a);
      r2=(-b-sqrt(det))/(2*a);
      printf("%lf %lf",r1,r2);
    }
  else if(m==0)
    {
      r1=-b/(2*a);
      printf("%lf",r1);
    }
  else
    printf("imaginary roots");
}
```

Test

Input

1 -8 -7

Output

8.795832 -0.79583

6 Distance between 2 points

Program Description

Write a program to compute the distance between two points. To read a point, the program should read 2 numbers from the user for the `x` and `y` coordinates. Hence your program should read numbers for the two points. Print the output on the stdout. Implement a function `distance(x1, y1, x2, y2)` that takes two points `(x1, y1)` and `(x2, y2)` as 4 parameters and returns the distance between the two points. Avoid duplicate calculations whereever possible.

Specification

Function `distance(x1,y1,x2,y2)` takes 4 parameters, the coordinates of 2 points in int and returns the distance between them using the formula ((x1-x2)^2 + (y1-y2)^2)^1/2 `float distance(int x1,int y1,int x2,int y2)`

Program Design

The program consists of the following.

1. `distance(x1,y1,x2,y2):` returns the distance between the points (x1,y1) and (x2,y2).
2. `main()`:inputs the two points, tests the function `distance(x1,y1,x2,y2)` and prints the result.

Program

```
#include<stdio.h>
#include<math.h>
float distance(int x1,int y1 ,int x2,int y2)
{
   float dis;
   dis=sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2));
   return dis;
}
int main()
{
   int x1,x2,y1,y2;
   float dis;
   scanf("%d %d %d %d",&x1,&y1,&x2,&y2);
   dis=distance(x1,y1,x2,y2);
   printf("%f",dis);
}
```

Test

Input

```
4 5 -3 3
```

Output

```
7.280110
```

7 Swap two variables.

Program Description

Initialize two variables with values read from the user and exchange (swap) their contents. Print them before and after the swap.

Program Design

The program consists of the following.

1. `main()`:inputs two numbers, a and b,prints them before and after swapping.

Program

```
#include <stdio.h>

int main ()
{
  int a;
  int b;

  scanf ("%d%d", &a, &b);
  printf ("a = %d, b = %d\n", a, b);
  int t = a;
  a = b;
  b = t;
  printf ("a = %d, b = %d\n", a, b);
  return 0;
}
```

```
a = 1    b = -3
a = -3   b = 1
```

8 Swap using function

Description

Define a function `swap()` to exchange the contents of the two variables, and check whether the function works as intended. If it does not work, what is the reason?

Specification

Function `swap(a,b)` takes two numbers as parameters and returns the numbers after swapping them.

Function

int swap(int a,int b) { int t; t=a; a=b; b=t; return (a,b);)

Reason

This function does not work because a function in C programming language can return only one value. It cannot return multiple values.

9 Circulate numbers

Program Description

Read four numbers `a, b, c, d` from stdin. Circulate them so that a gets the value of b, and so on: `a <- b <- c <- d <- a`

Program Design

The program consists of the following.

1. `main()`: inputs 4 numbers, circulates them in the way `a <- b <- c <- d <- a` and prints the numbers after circulation.

Algorithm

if there are n numbers from a_1 to a_n, a1=t; a1=a2; a2=a3; . . . $a_{(n-1)}=a_n$; a_n=t;

Program

```
#include <stdio.h>

int main ()
{
  int a, b, c, d,t;

  scanf ("%d%d%d%d", &a, &b, &c, &d);
  printf ("a = %d, b = %d, c = %d, d = %d\n", a, b, c, d);
  t = a;
  a = b;
  b = c;
  c = d;
  d = t;
  printf ("a = %d, b = %d, c = %d, d = %d\n", a, b, c, d);
  return 0;
}
```

```
a = 1    b = -3   c = 2   d = 6
a = -3   b = 2    c = 6   d = 1
```

10 Rearrange three numbers

Program Description

Read three numbers `a, b, c` from stdin. Write a program to rearrange them so that `a $\le$ b $\le$ c`.

Specification

Function `min2(a,b)` takes two integer parameters and returns the minimum of the two. Function `min3(a,b,c)` takes three integer parameters and returns the minimum of the three. `int min2(int a,int b) int min3(int a,int b,int c)`

Program Design

The program consists of the following.

1. `min2(a,b)`:returns minimum of two numbers.
2. `min3(a,b,c)`:returns minimum of three numbers.
3. `main()`:inputs three numbers,tests `min2(a,b)` and `min3(a,b)` and rearranges the three numbers a,b,c such that `a $\le$ b $\le$ c`.

Program

```
#include<stdio.h>
int min2(int a,int b)
{
   if(a<=b)
       return a;
   else
       return b;
}
int min3(int a, int b, int c)
{
   int temp;
   temp=min2(a,b);
   return min2(temp,c);
}
int main()
{
   int a,b,c,s,t,u;
   scanf("%d%d%d",&a,&b,&c);
   printf("a=%d, b=%d, c=%d\n",a,b,c);
   t=min3(a,b,c);
   if (t==a)
   {
       s=min2(b,c);
       if (s==b)
  u=c;
       else
  u=b;
   }
   else if (t==b)
{
    s=min2(a,c);
    if(s==a)
       u=c;
    else u=a;
}
else
{
    s=min2(a,b);
    if(s==a)
       u=b;
    else
       u=a;
}
   a=t;
   b=s;
   c=u;
   printf("a=%d, b=%d, c=%d",a,b,c);
}
```

```
a = 1    b = -3   c = 2
a = -3   b = 1    c = 2
```

11 Rearrange numbers in an array

Program Description

Fill an array of 3 numbers with numbers read from stdin. Write a program to rearrange them so that `a[0] $\le$ a[1] $\le$ a[2]`

Specification

Function `min2(a,b)` takes two integer parameters and returns the minimum of the two. Function `min3(a,b,c)` takes three integer parameters and returns the minimum of the three. `int min2(int a,int b) int min3(int a,int b,int c)`

Program Design

The program consists of the following.

1. `min2(a,b)`:returns minimum of two numbers.
2. `min3(a,b,c)`:returns minimum of three numbers.
3. `main()`:inputs three numbers and stores them in an array, tests `min2(a,b)` and `min3(a,b)` and rearranges the three numbers in the array such that `a[0]` $\le$ `a[1]` $\le$ `a[2]`.

Program

```
#include<stdio.h>
int min2(int a,int b)
{
   if(a<=b)
      return a;
   else
      return b;
}
int min3(int a, int b, int c)
{
   int temp;
   temp=min2(a,b);
   return min2(temp,c);
}
```

```
int main()
{
   int a[5],s,t,u,i;
   for (i=0;i<3;i++)
   {
      scanf("%d",&a[i]);
      printf("a[%d]=%d",i,a[i]);
      if (i==2)
  printf("\n");
      else
  printf(", ");
   }
   t=min3(a[0],a[1],a[2]);
   if (t==a[0])
   {
       s=min2(a[1],a[2]);
       if (s==a[1])
  u=a[2];
       else u=a[1];
   }
   else if (t==a[1])
{
   s=min2(a[0],a[2]);
   if(s==a[0])
      u=a[2];
   else u=a[0];
}
else
{
   s=min2(a[0],a[1]);
   if(s==a[0])
      u=a[1];
   else u=a[0];
}
   a[0]=t;
   a[1]=s;
   a[2]=u;
   printf("a[0]=%d, a[1]=%d, a[2]=%d",a[0],a[1],a[2]);
 }
```

a[1]= 1   a[2]= -3   a[3]= 2
a[1] = -3   a[2] = 1   a[3] = 2

[1]DEFINITION NOT FOUND.
[2]DEFINITION NOT FOUND.
[3]DEFINITION NOT FOUND.