Exercise 2: Expressions, Variables, Assignment

R Ram Kaushik

April 3, 2018

%

Assignment	1
Reg No	312217104125
Name	R Ram Kaushik
Grade	
Date	20-02-2018

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 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.

2.1 Specification

A function `area()`, which takes the radius `r` of the circle as input and returns the area, `perimeter()`, which takes the radius `r` of the circle as input and returns the perimeter.

2.2 Prototype

```
float area(float r)
float perimeter(float r)
```

2.3 Program Design

The program consists of the 2 functions `area(float r)` which finds the area of the circle, `perimeter(float r)` which finds the perimeter of the circle, and `main()` which reads the input from `stdin`, calls the functions and prints the result on `stdout`.

2.4 Algorithm

```
def area(r):
   return 3.14*r*r
def perimeter
   return 2*3.14*r
```

2.5 Source Code

```
#include<stdio.h>
float area(float r){
  return 3.14*r*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("%f %f %f",r,ar,p);
}
```

2.6 Test Input

```
15
```

2.7 Output

```
15.000000    706.500000    94.199997
```

3 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()`.

3.1 Specification

A function `is_leap()`, which takes the `year` as input and returns either `true` or `false` in bool.

3.2 Prototype

```
bool is_leap(int year)
```

3.3 Program Design

The program consists of a function `is_leap(int year)` which returns `true` if the year is leap and returns `false` if the year is not leap and `main()` which gets the input from `stdin`,calls the function and prints the value on `stdout`.

3.4 Algorithm

```
def is_leap(year):
   if year%4==0 and year%100!=0 or year%400==0:
      return true
   else:
      return false
```

3.5 Source Code

```
#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,leap;
  while(scanf("%d",&year)!=EOF){
    leap=is_leap(year);
    printf("%d\n",leap);
  }
}
```

3.6 Test Input

```
2009
2000
1900
1936
```

3.7 Output

```
0
1
0
1
```

4 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 wherever possible.

4.1 Specification

A function `sign()`, which takes `n` as the input and returns the sign of it.

4.2 Prototype

```
int sign(int n)
```

4.3 Program Design

The program consists of the function `sign(int n)` which returns the sign based on the number, and `main()` which gets the input from `stdin`,calls the function `sign(n)` and prints the result on `stdout`.

4.4 Algorithm

```
def sign(n):
   if n>0:
      return 1
   elif n==0:
      return 0
   return -1
roots=(-b+d)/2a,(-b-d)/2a
```

4.5 Source Code

```
#include<stdio.h>
#include<math.h>
int sign(int n){
  if (n>0){
    return 1;
  }
  else if(n==0){
    return 0;
  }
  else{
    return -1;
  }
}
int main(){
  int a,b,c,m,det;
  float r1,r2;
  while(scanf("%d%d%d",&a,&b,&c)!=EOF){
    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("%f %f",r1,r2);
    }
    else if(m==0){
      r1=-b/(2*a);
      printf("%f",r1);
    }
    else{
      printf("imaginary roots");
    }
  }
}
```

4.6 Test Input

```
1 -2 1
1 -3 2
1 1 1
```

4.7 Output

```
1.000000
2.000000

1.000000
imaginary roots
```

5 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 wherever possible.

5.1 Specification

A function `distance()` which takes the coordinates of 2 points as inputsin int and returns the distance between them.

5.2 Prototype

```
float distance(int x1,int y1,int x2,int y2)
```

5.3 Program Design

The program consists of the functions `distance(int x1,int y1,int x2,int y2)` which returns the distance between the points (x1,y1) and (x2,y2) and `main()` which gets the inputs from `stdin`, calls the function and prints the result on `stdout`.

5.4 Algorithm

```
def distance(x1,y1,x2,y2):
   d=sqrt(pow(x1-x2,2)+pow(y1-y2,2))
   return d
```

5.5 Source Code

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

5.6 Test Input

```
5 6
1 2
```

5.7 Output

5.656854

6 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.

6.1 Specification

A function which swaps the 2 numbers.

6.2 Program Design

The program consists of `main()` which gets the input from `stdin`, swaps them and prints them on `stdout`.

6.3 Algorithm

```
t=a
a=b
b=t
```

6.4 Source Code

```
#include <stdio.h>
int main (){
  int a,b,t;
  scanf ("%d%d", &a, &b);
  t = a;
  a = b;
  b = t;
  printf ("%d %d\n", a, b);
  return 0;
}
```

6.5 Test Input

```
5 10
```

6.6 Output

```
10 5
```

7 Swap using function

*Problem 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?

7.1 Specification

A function `swap()` takes two numbers as inputs and returns the numbers after swapping them.

7.2 Prototype

```
int swap(int* a, int* b)
```

7.3 Program Description

The program contains a function `swap(int* a, int* b)`, which swaps the numbers and `main()` which gets the input from `stdin`, calls the function and prints the output on `stdout`

7.4 Algorithm

```
def swap(a, b):
   a,b=b,a
```

7.5 Source Code

```
#include<stdio.h>
void swap(int* a,int* b){
  int t=*a;
  *a=*b;
  *b=t;
}
int main(){
  int a,b;
  scanf("%d%d",&a,&b);
  swap(&a,&b);
  printf("%d %d\n",a,b);
}
```

7.6 Test Input

```
5 10
```

7.7 Output

```
10 5
```

8 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`

8.1 Specification

A function that circulates the numbers.

8.2 Program Design

The program consists of `main()` which gets the input from `stdin`, circulates them in the way `a <- b <- c <- d <- a` and prints the numbers on `stdout`.

8.3 Algorithm

```
t=a1;
a1=a2;
a2=a3;
.
.
.
a_(n-1)=a_n;
a_n=t;
```

8.4 Source Code

```
#include <stdio.h>
int main (){
  int a, b, c, d,t;
  scanf ("%d%d%d%d", &a, &b, &c, &d);
  t = a;
  a = b;
  b = c;
  c = d;
  d = t;
  printf ("%d %d %d %d\n", a, b, c, d);
  return 0;
}
```

8.5 Test Input

```
1 -3 2 6
```

8.6 Output

```
-3 2 6 1
```

9 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`.

9.1 Specification

2 functions `min2()`, which takes two integers as input and returns the minimum of the two and `min3()`, which takes three integers as inputs and returns the minimum of the three.

9.2 Prototype

```
int min2(int a,int b)
int min3(int a,int b,int c)
```

9.3 Program Design

The program consists of the functions `min2(a,b)` which returns minimum of two numbers, `min3(a,b,c)` which returns minimum of three numbers and `main()` which gets inputs from `stdin`, calls the function and prints the result on `stdout`.

9.4 Algorithm

```
def min2(a,b):
   if a<=b:
      return a
   return b
def min3(a,b,c):
   t=min2(a,b)
   return min2(t,c)
```

9.5 Source Code

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

9.6 Test Input

```
1 -3 2
```

9.7 Output

```
-3 1 2
```

10 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]`

10.1 Specification

2 functions `min2()`, which takes two integers as input and returns the minimum of the two and `min3()`, which takes three integers as inputs and returns the minimum of the three.

10.2 Prototype

```
int min2(int a,int b)
int min3(int a,int b,int c)
```

10.3 Program Design

The program consists of the functions `min2(a,b)` which returns minimum of two numbers, `min3(a,b,c)` which returns minimum of three numbers and `main()` which gets inputs from `stdin`, calls the function and prints the result on `stdout`.

10.4 Algorithm

```
def min2(a,b):
   if a<=b:
      return a
   return b
def min3(a,b,c):
   t=min2(a,b)
   return min2(t,c)
```

10.5 Source Code

```
#include<stdio.h>
int swap(int* a,int* b){
   int t=*a;
   *a=*b;
   *b=t;
}
int main(){
   int a[5],s=0,p;
   for(int i=0;i<3;i++){
      scanf("%d",&a[i]);
      s+=a[i];
   }
   if(a[0]>a[1]){
     swap(&a[0],&a[1]);
   }
   if(a[1]>a[2]){
     swap(&a[1],&a[2]);
   }
   if(a[0]>a[1]){
     swap(&a[0],&a[1]);
   }
   for(int i=0;i<3;i++){
      printf("%d ",a[i]);
   }
}
```

10.6 Test Input

```
23 52 13
```

10.7 Output

```
13 23 52
```