Jaypee Institute of information Technology, Noida

Software Development Fundamentals 1 - Lab Sheet

Week: 29 Oct- 24th Nov, 2018

Assignment Type: Practice

Lab A & B (Functions)

Instructions to be followed while carrying out the Lab:

1. Always save your lab work and keep backup of files

2. Perform the all the experiments

Q1. Visit the specified links to perform these lab experiments:

For Experiment 1:

Description: Demonstration of functions and to call them with proper arguments

Link: http://cse02-iiith.vlabs.ac.in/exp2/simulation/index.html

Procedure

1. Click on the square to define a function for calculating the area of a square.
2. Similarly define functions for the other geometrical figures
3. The defined functions are shown in the middle window
4. Now do appropriate function calls in the main program to compute the area of the figure displayed
5. Press execute to execute the code and see the output

Experiment:

1. Click on the square to define a function for calculating the area of a square.

2. Similarly define functions for the other geometrical figures.

3. The defined functions are shown in the middle window.

4. Now make appropriate function calls in the main program to compute the area of the figure displayed.

5. Press execute to execute the code and see the output

Execute the experiment with different inputs and analyze the output.

For Experiment 2:

Description: Demonstration of how to solve the classical puzzle of Towers of Hanoi through recursion with simulation

Link: http://cse02-iiith.vlabs.ac.in/exp9/simulation/index.html

Procedure

1. Select the value of N (values must me greater than 1 and less than or equal to 5).
2. Press next to see the execution of the code.
3. Relevant line in the code is shown here.

The output of the code is shown in the right

Execute the experiment with different values of N and analyze the output.

Q2: WAP in C to display all prime numbers between two Intervals using function.

Q3: WAP in C to check Prime and Armstrong Number by making function.

Q4: WAP in C to check whether a number can be expressed as the sum of two prime number

Q5: WAP in C to find sum of natural numbers using recursion

Q6: WAP in C to calculate factorial of a number using recursion

Q7: WAP in C to find G.C.D using recursion

Q8: WAP in C to reverse a sentence using recursion

Q9: WAP in C to calculate the power of a number using recursion

Q10: WAP in C to convert binary number to decimal and vice-versa using function.

Q11: WAP in C to convert octal Number to decimal and vice-versa using function.

Q12: WAP in C to convert binary number to octal and vice-versa using function.

Q13: Write a program in C to find the sum of the series 1!/1+2!/2+3!/3+4!/4+5!/5 using the function.

Q14: Write a program in C to get the largest element of an array using the function.

Q15: Write a program in C to print all perfect numbers in given range using the function.

Q16: Write a program in C to calculate the sum of numbers from 1 to n using recursion.

Q17: Write a program in C to Print Fibonacci Series using recursion.

Q18: Write a program in C to find the Factorial of a number using recursion.

Q19: Write a program in C to find the LCM of two numbers using recursion.

Q20: Write a program in C to multiply two matrix using recursion.

Q21: Write a program in C to calculate the power of any number using recursion.

Q22: Write a program in C for binary search using recursion

Q23: Write a program in C to Reverse a Sentence Using Recursion.

Q24: Write a C program to create a structure student, containing name and roll and display the information using function.

Q25: Write a C program to add two distances (feet-inch system) and display the result using structures and functions.

Q26: Write a C Program to Add Two Complex Numbers by Passing Structure to a Function

Q27: Write a C Program to Calculate Difference Between Two Time Periods using structure and function.

Q28: Write a C Program to Multiply two Matrices by Passing Matrix to a Function

Q29: Execute the following codes and interpret the output:

1.
```
#include <stdio.h>
int var = 20;
int main()
{
   int var = var;
   printf("%d ", var);
   return 0;
}
```
2.
```
#include <stdio.h>
int main()
{
 int x = 1, y = 2, z = 3;
 printf(" x = %d, y = %d, z = %d n", x, y, z);
 {
     int x = 10;
     float y = 20;
     printf(" x = %d, y = %f, z = %d n", x, y, z);
     {
          int z = 100;
          printf(" x = %d, y = %f, z = %d n", x, y, z);
     }
 }
 return 0;
}
```
3.
```
#include <stdio.h>
main( )
{
int i;
```

```
for ( i=0; i<5; i++ )
{
  int i = 10;
  printf ( "%d ", i );
  i++;
}
return 0;
}
```

4.
```
#include <stdio.h>
int main()
{
static int i=5;
if(--i){
main();
printf("%d ",i);
}
}
```

5.
```
#include <stdio.h>


int fun()
{
 static int num = 16;
 return num--;
}

int main()
{
 for(fun(); fun(); fun())
   printf("%d ", fun());
 return 0;
}
```

6.
```
#include <stdio.h>
int a, b, c = 0;
void prtFun (void);
int main ()
{
   static int a = 1; /* line 1 */
   prtFun();
   a += 1;
   prtFun();
   printf ( "n %d %d " , a, b) ;
}

void prtFun (void)
{
   static int a = 2; /* line 2 */
   int b = 1;
   a += ++b;
   printf (" n %d %d " , a, b);
}
```

7.
```
#include <stdio.h>
int fun(int n)
{
```

```
    static int s = 0;
    s = s + n;
    return (s);
}

int main()
{
    int i = 10, x;
    while (i > 0)
    {
        x = fun(i);
        i--;
    }
    printf ("%d ", x);
    return 0;
}
```

8.
```
void foo(int n, int sum)
{
 int k = 0, j = 0;
 if (n == 0) return;
   k = n % 10;
 j = n / 10;
 sum = sum + k;
 foo (j, sum);
 printf ("%d,", k);
}

int main ()
{
 int a = 2048, sum = 0;
 foo (a, sum);
 printf ("%dn", sum);

 getchar();
}
```

9.
```
#include <stdio.h>
int funcf (int x);
int funcg (int y);

main()
{
    int x = 5, y = 10, count;
    for (count = 1; count <= 2; ++count)
    {
        y += funcf(x) + funcg(x);
        printf ("%d ", y);
    }
}

funcf(int x)
{
    int y;
    y = funcg(x);
    return (y);
}
```

```
funcg(int x)
{
   static int y = 10;
   y += 1;
   return (y+x);
}
```

10.
```
#include<stdio.h>
int f(int n, int k)
{
   if (n == 0)
      return 0;
   else if (n % 2)
      return f(n/2, 2*k) + k;
   else return f(n/2, 2*k) - k;
}
int main ()
{
   printf("%d", f(20, 1));
   return 0;
}
```

Q30: Understand and analyze the output of the following codes

1. **Passing Structure to Function**

```
#include<stdio.h>
#include<conio.h>
//-------------------------------------
struct Example
{
  int num1;
  int num2;
}s[3];
//-------------------------------------
void accept(struct Example *sptr)
{
  printf("\nEnter num1 : ");
  scanf("%d",&sptr->num1);
  printf("\nEnter num2 : ");
  scanf("%d",&sptr->num2);
}
//-------------------------------------
void print(struct Example *sptr)
{
  printf("\nNum1 : %d",sptr->num1);
  printf("\nNum2 : %d",sptr->num2);
}
//-------------------------------------
void main()
{
int i;
clrscr();
for(i=0;i<3;i++)
accept(&s[i]);

for(i=0;i<3;i++)
```

```
print(&s[i]);

getch();
}
```

2. **C program to pass a single element of an array to function**

```
#include <stdio.h>
```

```
void display(int age)
{
   printf("%d", age);
}


int main()
{
   int ageArray[] = { 2, 3, 4 };
   display(ageArray[2]); //Passing array element ageArray[2] only.
   return 0;
}
```

3. **C program to pass an array containing age of person to a function. This function should find average age and display the average age in main function.**

```
#include <stdio.h>
float average(float age[]);

int main()
{
        float avg, age[] = { 23.4, 55, 22.6, 3, 40.5, 18};
        avg = average(age); // Only name of an array is passed as an argument
        printf("Average age = %.2f", avg);
        return 0;
}

float average(float age[])
{
        int i;
        float avg, sum = 0.0;
        for (i = 0; i < 6; ++i) {
                sum += age[i];
        }
        avg = (sum / 6);
```

```
        return avg;
}
```

4. Pass two-dimensional arrays to a function

```
#include <stdio.h>
void displayNumbers(int num[2][2]);
int main()
{
   int num[2][2], i, j;
   printf("Enter 4 numbers:\n");
   for (i = 0; i < 2; ++i)
      for (j = 0; j < 2; ++j)
         scanf("%d", &num[i][j]);
   // passing multi-dimensional array to displayNumbers function
   displayNumbers(num);
   return 0;
}

void displayNumbers(int num[2][2])
{
     int i, j;
   printf("Displaying:\n");
   for (i = 0; i < 2; ++i)
      for (j = 0; j < 2; ++j)
         printf("%d\n", num[i][j]);
}
```