Q1.

#include<stdio.h>

void fun(int x)

{

if(x > 0)

{

fun(--x);

printf("%d\t", x);

fun(--x);

}

}

int main()

{

int a = 4;

fun(a);

getchar();

return 0;

}

**Output: 0 1 2 0 3 0 1**

Q2.

int fun(int a[],int n)

{

int x;

if(n == 1)

return a[0];

else

x = fun(a, n-1);

if(x > a[n-1])

return x;

else

return a[n-1];

}

int main()

{

int arr[] = {12, 10, 30, 50, 100};

printf(" %d ", fun(arr, 5));

getchar();

return 0;

}

**Output: 100**

Q4.

int fun(int i)

{

if ( i%2 ) return (i++);

else return fun(fun( i - 1 ));

}

int main()

{

printf(" %d ", fun(200));

getchar();

return 0;

}

**Output: 199**

Q5.

int fun1(int n)

{

if(n == 1)

return 0;

else

return 1 + fun1(n/2);

}

**Answer: The function calculates and returns log2floor. For example, if n is between 8 and 15 then fun1() returns 3. If n is between 16 to 31 then fun1() returns 4.**

Q6.

/\* Assume that n is greater than or equal to 0 \*/

void fun2(int n)

{

if(n == 0)

return;

fun2(n/2);

printf("%d", n%2);

}

**Answer: The function fun2() prints binary equivalent of n. For example, if n is 21 then fun2() prints 10101.**

Q7.

#include <stdio.h>

int fun ( int n, int \*fp )

{

int t, f;

if ( n <= 1 )

{

\*fp = 1;

return 1;

}

t = fun ( n-1, fp );

f = t + \*fp;

\*fp = t;

return f;

}

int main()

{

int x = 15;

printf("%d\n",fun(5, &x));

return 0;

}

**Output:8**

The program calculates nth Fibonacci Number. The statement t = fun ( n-1, fp ) gives the (n-1)th Fibonacci number and \*fp is used to store the (n-2)th Fibonacci Number. Initial value of \*fp (which is 15 in the above prgram) doesn’t matter. Following recursion tree shows all steps from 1 to 10, for exceution of fun(5, &x).

Q9.

#include <stdio.h>

void fun(int n)

{

if(n > 0)

{

fun(n-1);

printf("%d ", n);

fun(n-1);

}

}

int main()

{

fun(4);

return 0;

}

**Output:1 2 1 3 1 2 1 4 1 2 1 3 1 2 1**

Q10.

#include<stdio.h>

void fun(int\*, int\*);

int main()

{

int i=5, j=2;

fun(&i, &j);

printf("%d, %d", i, j);

return 0;

}

void fun(int \*i, int \*j)

{

\*i = \*i\*\*i;

\*j = \*j\*\*j;

}

**Output: 25, 4**

Q11.

#include<stdio.h>

int reverse(int);

int main()

{

int no=5;

reverse(no);

return 0;

}

int reverse(int no)

{

if(no == 0)

return 0;

else

printf("%d,", no);

reverse (no--);

}

Q12.

#include<stdio.h>

int main()

{

void fun(char\*);

char a[100];

a[0] = 'A'; a[1] = 'B';

a[2] = 'C'; a[3] = 'D';

fun(&a[0]);

return 0;

}

void fun(char \*a)

{

a++;

printf("%c", \*a);

a++;

printf("%c", \*a);

}

**Output: BC**

Q13.

#include<stdio.h>

int main()

{

int fun(int);

int i = fun(10);

printf("%d\n", --i);

return 0;

}

int fun(int i)

{

return (i++);

}

**Output: 9**

Q14.

#include<stdio.h>

int check (int, int);

int main()

{

int c;

c = check(10, 20);

printf("c=%d\n", c);

return 0;

}

int check(int i, int j)

{

int \*p, \*q;

p=&i;

q=&j;

return i>=45 ? \*p: (\*q);

}

**Output: C= 20**

Q15.

#include<stdio.h>

int main()

{

int i=1;

if(!i)

printf("IndiaBIX %d,", i);

else

{

i=0;

printf("C-Program %d", i);

}

return 0;

}

**output: C-Program 0**

Q16.

#include<stdio.h>

int addmult(int ii, int jj)

{

int kk, ll;

kk = ii + jj;

ll = ii \* jj;

return (kk, ll);

}

int main()

{

int i=3, j=4, k, l;

k = addmult(i, j);

l = addmult(i, j);

printf("%d %d\n", k, l);

return 0;

}

**output: 12 12**

Q17.

#include<stdio.h>

int func1(int);

int main()

{

int k=35;

k = func1(k=func1(k=func1(k)));

printf("k=%d\n", k);

return 0;

}

int func1(int k)

{

k++;

return k;

}

**output: K= 38**

Q18.

#include<stdio.h>

int check(int);

int main()

{

int i=45, c;

c = check(i);

printf("%d\n", c);

return 0;

}

int check(int ch)

{

if(ch >= 45)

return 100;

else

return 10;

}

**output: 100**

Q19.

#include <stdio.h>

void fun(char\*\*);

int main()

{

char \*argv[] = {"ab", "cd", "ef", "gh"};

fun(argv);

return 0;

}

void fun(char \*\*p)

{

char \*t;

t = (p+= sizeof(int))[-1];

printf("%s\n", t);

}

**output: cd**

Q20.

#include<stdio.h>

int fun(int i)

{

i++;

return i;

}

int main()

{

int fun(int);

int i=3;

fun(i=fun(fun(i)));

printf("%d\n", i);

return 0;

}

**output: 5**

Q21.

#include<stdio.h>

int main()

{

int a[3][4] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 };

printf("%u, %u, %u\n", a[0]+1, \*(a[0]+1), \*(\*(a+0)+1));

return 0;

}

**output: 1006, 2, 2**

Q22.

#include<stdio.h>

int main()

{

int arr[3] = {2, 3, 4};

char \*p;

p = arr;

p = (char\*)((int\*)(p));

printf("%d, ", \*p);

p = (int\*)(p+1);

printf("%d", \*p);

return 0;

}

**output: 2,0**

Q23.

#include<stdio.h>

int main()

{

char \*str;

str = "%d\n";

str++;

str++;

printf(str-2, 300);

return 0;

}

**output: 300**

Q24.

#include<stdio.h>

int main()

{

char str[] = "peace";

char \*s = str;

printf("%s\n", s++ +3);

return 0;

}

**output: ce**

Q25.

#include<stdio.h>

int main()

{

char \*p;

p="hello";

printf("%c\n", \*\*&\*&p);

return 0;

}

**output: h**

Q26.

#include<stdio.h>

int func(int\*\*);

int main()

{

int a=5, \*aa; /\* Address of 'a' is 1000 \*/

aa = &a;

a = func(&aa);

printf("%d\n", a);

return 0;

}

int func(int \*\*ptr)

{

int b;

b = \*\*ptr\*\*\*ptr;

return (b);

}

**output: 25**

Q27.

#include<stdio.h>

#include<string.h>

int main()

{

int i, n;

char \*x="Alice";

n = strlen(x);

\*x = x[n];

for(i=0; i<=n; i++)

{

printf("%s ", x);

x++;

}

printf("\n", x);

return 0;

}

**output: lice ice ce e**

Q28.

#include<stdio.h>

int main()

{

int i, a[] = {2, 4, 6, 8, 10};

change(a, 5);

for(i=0; i<=4; i++)

printf("%d, ", a[i]);

return 0;

}

void change(int \*b, int n)

{

int i;

for(i=0; i<n; i++)

\*(b+1) = \*(b+i)+5;

}

**output: 2, 15, 6, 8, 10**

Q29.

#include<stdio.h>

int main()

{

int arr[] = {12, 13, 14, 15, 16};

printf("%d, %d, %d\n", sizeof(arr), sizeof(\*arr), sizeof(arr[0]));

return 0;

}

**output: 20, 4, 4**

Q30.

#include <stdio.h>

void main()

{

char \*s= "hello";

char \*p = s;

printf("%c\t%c", \*(p + 3), s[1]);

}

**output: l e**

Q31.

#include <stdio.h>

void main()

{

char \*s= "hello";

char \*p = s;

printf("%c\t%c", 1[p], s[1]);

}

**output: e e**

Q32.

#include <stdio.h>

void foo( int[] );

int main()

{

int ary[4] = {1, 2, 3, 4};

foo(ary);

printf("%d ", ary[0]);

}

void foo(int \*p)

{

int i = 10;

\*p = i;

printf("%d ", p[0]);

}

**output: 10 10**

Q33.

#include <stdio.h>

int main()

{

int ary[4] = {1, 2, 3, 4};

int \*p = ary + 3;

printf("%d\n", p[-2]);

}

**output: 2**

Q34.

#include <stdio.h>

int main()

{

int ary[4] = {1, 2, 3, 4};

int \*p = ary + 3;

printf("%d %d\n", p[-2], ary[\*p-3]);

}

**output: 2 1**

Q35.

#include <stdio.h>

int main()

{

char \*str="IncludeHelp";

printf("%c\n",\*&\*str);

return 0;

}

**output: I**

Q36.

#include <stdio.h>

char\* strFun(void)

{

char \*str="IncludeHelp";

return str;

}

int main()

{

char \*x;

x=strFun();

printf("str value = %s",x);

return 0;

}

**output: str value= IncludeHelp**

Q37.

#include <stdio.h>

int main()

{

char ch=10;

char \*ptr=&ch;

printf("%d,%d",\*(char\*)ptr,++(\*(char\*)ptr));

return 0;

}

**output: 11, 11**

Q38.

#include <stdio.h>

void fun(int \*ptr)

{

int a =10;

ptr= &a;

\*ptr = \*ptr + 200;

}

int main()

{

int num=50;

int \*pp=&num;

fun(&\*pp);

printf("%d,%d",num,\*pp);

return 0;

}

**output: 50, 50**

Q39.

#include <stdio.h>

void fun(int \*\*ptr)

{

\*\*ptr=100;

}

int main()

{

int num=50;

int \*pp=&num;

fun(&pp);

printf("%d,%d",num,\*pp);

return 0;

}

**output: 100, 100**