Result: **Pass**
Marks: **30/30**
Percentage: **100.00 %**

Questions: **10**
Correct Answers: **10**
Attempted: **10**

1. Consider the following recursive implementation to find the nth fibonacci number:

```
int fibo(int n)
{
     if(n == 1)
        return 0;
     else if(n == 2)
        return 1;
     return ________;
}
int main()
{
     int n = 5;
     int ans = fibo(n);
     printf("%d",ans);
     return 0;
}
```

Answers

1. fibo(n - 1)

2. fibo(n - 1) + fibo(n - 2)

3. fibo(n) + fibo(n - 1)

4. fibo(n - 2) + fibo(n - 1)

2. What will be the output of the C program?

```
#include<stdio.h>
void function();
int main()
{
        function();
        return  0;
}
void function()
{
        printf("Function in C is awesome");
}
```

Answers

1. Function in C is awesome

2. No output

3. Runtime error

4. Compiler error

3. What does the following fun() do in general?

```
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;
}
```

Answers

1. 130

2. 120

3. 100

4.
```
void main()
    {
        static int x = 3;
        x++;
        if (x <= 5)
        {
            printf(" hi ");
            main();
        }
    }
```

Answers

1. hi hi hi

2. hi

3. Infinite hi

4. hi hi

5. What does the following fun() do in general?

```
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;
}
```

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

Answers

1. 6

2. 5

3. 3

4. 8

6. Can we use a function as a parameter of another function? [Eg: void wow(int func())]

Answers

1. Yes, and we can use the function value conveniently

2. Yes, but we call the function again to get the value, not as convenient as in using variable

3. No, C does not support it.

4. This case is compiler dependent

7. Which of the following statement are correct? (I) The maximum value a variable can hold depends upon its storage class.

Answers

1. Only II is correct

2. Both I & II are correct

3. Only I is correct

4. Both I & II are incorrect

8. What is the output of C Program with functions and pointers.?

```
void myshow(int *);

void main()
{
   int a=10;
   printf("%d ", a);
   myshow(&a);
   printf("%d", a);

}

void myshow(int *k)
{
   *k=20;
}
```

Answers

1. 10 10

2. 10 20

3. 20 20

4. Compiler error

9. What will be the output of the following program?

```
#include<stdio.h>
void fun()
{
            auto int I = 1;
            register char a = 'D';
            static int p = 0;
            printf("%d %d %d", I, a, p);
}
int main()
{
          fun();
         return 0;
}
```

Answers

1. 1 D 0

2. 1 0 0

3. 0 D 1

4. 1 68 0

10. The default parameter passing mechanism is

Answers

1. call by value

2. call by reference

3. call by value result

4. none of above