Introduction to Computer Science and C Programming-Midterm Exam Answer

1. Greatest Common Divisor(GCD)

```
int GCD_recursive(int A,int B)
{
	if(___(1)___)
		return B;
	else
		return___(2)___;
}
int GCD(int A,int B)
{
	while(___(3)___)
	{
		if(A>B)
			___(4)___;
		else
			___(5)___;
	}
	if(___(6)___)
		return B;
	else
		return A;
}
int main()
{
	int A,B;
	scanf("%d",&A);
	scanf("%d",&B);
	printf("GCD_recursive=%d\n",GCD_recursive(A,B));
	printf("GCD=%d\n",GCD(A,B));
	system("pause");
	return 0;
}
```

Above is a program that implement *Greatest Common Divisor (GCD) method in two ways.* Please write down the correct statement of the blanket. Here is an example of this program:
input: 27 45 , output: GCD_recursive=9 GCD=9

Ans:

(1) A % B == 0

(2) GCD_recursive(B , A%B)

(3) A != 0 && B != 0

(4) A %= B

(5) B %= A

(6) A == 0

2. Please write the output of this program.

```
#include <stdio.h>
int calculator(int x,int y, int select)
{
        static int output = 0;
        switch(select)
        {
                case 0:
                        output+=(x+y);
                        break;
                case 1:
                        output+=(x-y);
                        break;
                case 2:
                        output+=(x*y);
                        break;
                case 3:
                        output+=(x/y);
                        break;
                default:
                        output=(x+y)+(x-y)+(x*y)+(x/y);
                        break;
        }
        return output;
}
int main(){
        int x=3,y=2;
        printf("Add:    %d\n", calculator( x, y, 0));
        printf("Sub:    %d\n", calculator( x, y, 1));
        printf("Mul:    %d\n", calculator( x, y, 2));
        printf("Div:    %d\n", calculator( x, y, 3));
        printf("Other:%d\n", calculator( x, y, 4));
        return 0;
}
```

Ans:

```
        Add:5
        Sub:6
        Mul:12
        Div:13
        Other:13
```

3. Please write the output of this program.

```c
#include <stdio.h>
int* fun1(int *n1, int *n2);

int* fun1(int *n1, int *n2){
	if(*n1 > *n2){
		*n1 -= *n2;
		return fun1(n2, n1);
	}
	return n1;
}

int main() {
	int n1 = 28, n2 = 15;
	int *p, *q;
	p = fun1(&n1, &n2);
	q = fun1(&n1, fun1(&n1, p));

	printf("*p = %d\n*q = %d\nn1 = %d\nn2 = %d\n", *p, *q, n1, n2);
	return 0;
}
```

Ans:

```
*p = 2
*q = 2
n1 = 7
n2 = 2
```

4. Please write the output of this program.

```c
int Ackermans(int m,int n)
{
	if(m==0)
		return n+1;
	else if(n==0)
		return Ackermans(m-1,1);
	else
		return Ackermans(m-1,Ackermans(m,n-1));
}
```

I. Ackermans(0,4): 5

II. Ackermans(2,0): 3

III. Ackermans(2,2): 7

IV. Ackermans(2,1): 5

V. Ackermans(1,2): 4

5. Please write the output of this program.

```
#include <stdio.h>
void fun1(int *num1, int *num2) {
	int tmp = *num1;
	*num1 = *num2;
	*num2 = tmp;
}
int* fun2(int *num1, int *num2) {
	*num1 += *num2;
	return num1;
}
int fun3(int num1, int num2) {
	num1 += num2;
	return num1;
}
int main() {
	int *p, *q, r = 23, s = 99, t = 10;
	p = &r;
	q = p;
	*p = 90;
	p = &s;
	t += *q;
	printf("%d, %d, %d, %d, %d\n", *p, *q, r, s, t);

	r = 4; s = 3; t = 2;
	p = &s; q = &r;
	fun1(p, q);
	fun2(p, q);
	fun3(s, t);
	printf("%d, %d, %d, %d, %d\n", *p, *q, r, s, t);

	r = 11; s = 22; t = 43;
	p = q = &t;
	*p += 2;
	*q += 1;
	if(p == q) r += 3;
	s = *fun2(&r, p);
	printf("%d, %d, %d, %d, %d\n", *p, *q, r, s, t);

	r = 4; s = 1; t = 10;
	p = &t; q = &r;
	*q += *p;
	t = fun3(*fun2(q, &r), *p);
	printf("%d, %d, %d, %d, %d\n", *p, *q, r, s, t);
	return 0;
}
```

Ans:

```
99, 90, 90, 99, 100
7, 3, 3, 7, 2
46, 46, 60, 60, 46
38, 28, 28, 1, 38
```

6. Gnome sort.

```
#include<stdio.h>
#define SIZE 4
void print_arr(int arr[])
{
	int i;
	for(i=0;i<SIZE;i++)
		printf("%d ", arr[i]);
	printf("\n");
}
int main(void)
{
	int arr[SIZE] = {11, 5, 2015};
	int index = 0;
	print_arr(arr);
	while(index < SIZE)
	{
		if(index == 0)
		{
			index++;
		}
		else
		{
			if(arr[index] < arr[index-1])
			{
				int tmp = arr[index];
				arr[index] = arr[index-1];
				arr[index-1] = tmp;
				index--;
			}
			else
			{
				index++;
			}
		}
		print_arr(arr);
	}
	return 0;
}
```

The code above is a simple sorting algorithm, called Gnome sort. Please trace the program and write down what will show on the screen. You will get 0 point even if there is only one mistake. For example, if arr is {2,4,3,1}. Your answer should be:

2 4 3 1 → 2 4 3 1 → 2 4 3 1 → 2 3 4 1 → 2 3 4 1 → 2 3 4 1 → 2 3 1 4 →
2 1 3 4 → 1 2 3 4 → 1 2 3 4 → 1 2 3 4 → 1 2 3 4 → 1 2 3 4

ANS:
11 5 2015 0 → 11 5 2015 0 → 5 11 2015 0 → 5 11 2015 0 → 5 11 2015 0 →
5 11 2015 0 → 5 11 0 2015 → 5 0 11 2015→ 0 5 11 2015 → 0 5 11 2015 →
0 5 11 2015 → 0 5 11 2015 → 0 5 11 2015

7. A stack is a special data structure, which could be simulated by using array and contains only two operation: PUSH, POP. Now, there is an empty stack. We PUSH A, B, C, D, E into stack in order. However, during the process when we PUSH them, we also use some POP operations(randomly) and output the result. In the following options, which is the possible output (may be one or more than one answers)? __________________

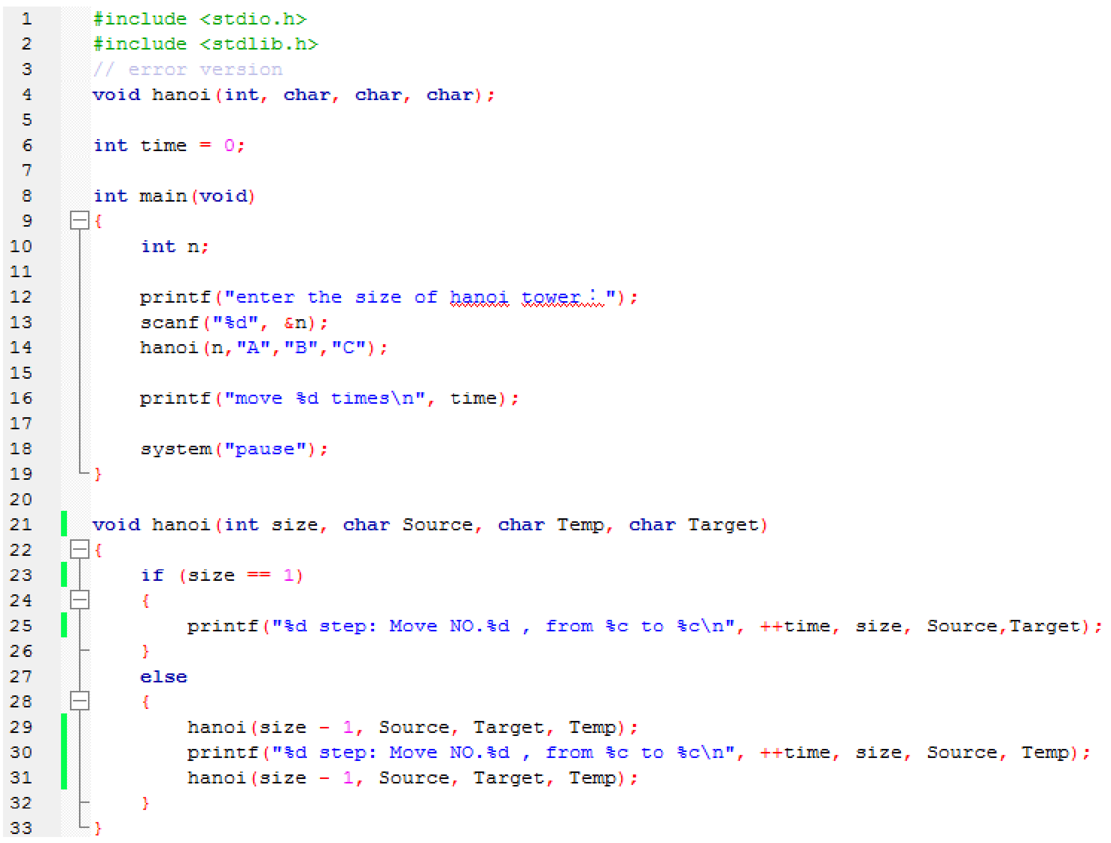
(A) ABCDE (B) EABCD (C) ABDEC (D) CBADE (E) BECDA

ANS: ACD

A	B	C	D	E
PUSH(A)	PUSH(A)	PUSH(A)	PUSH(A)	PUSH(A)
POP() → A	PUSH(B)	POP() → A	PUSH(B)	PUSH(B)
PUSH(B)	PUSH(C)	PUSH(B)	PUSH(C)	POP() → B
POP() → B	PUSH(D)	POP() → B	POP() → C	PUSH(C)
PUSH(C)	PUSH(E)	PUSH(C)	POP() → B	PUSH(D)
POP() → C	POP() → E	PUSH(D)	POP() → A	PUSH(E)
PUSH(D)	POP() → D	POP() → D	PUSH(D)	POP() → E
POP() → D		PUSH(E)	POP() → D	POP() → D
PUSH(E)		POP() → E	PUSH(E)	
POP() → E		POP() → C	POP() → E	

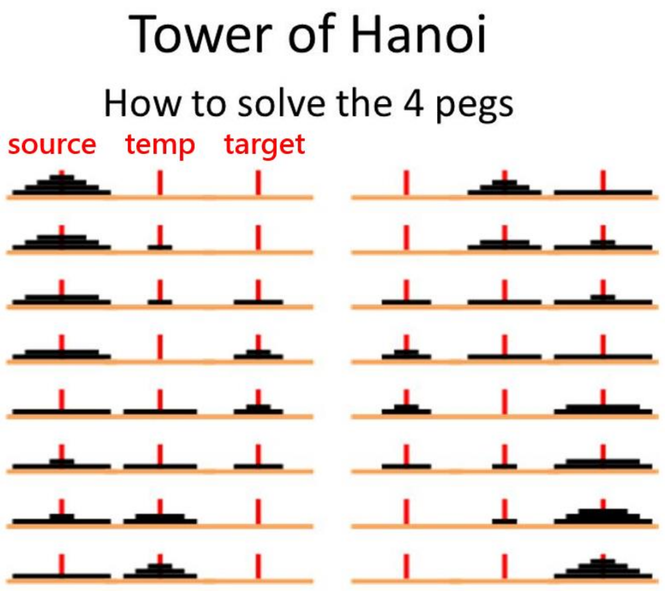
8. Tower of Hanoi

Tower of Hanoi is a classic mathematical game, it consist of three rods and disks. **The puzzle starts with the disks in a neat stack in ascending order of size on the first rod, the smallest at the top. The goal is to move the entire stack to the third rod.** Each step should follow these rules:



```
1  #include <stdio.h>
2  #include <stdlib.h>
3  // error version
4  void hanoi(int, char, char, char);
5
6  int time = 0;
7
8  int main(void)
9  {
10     int n;
11
12     printf("enter the size of hanoi tower :");
13     scanf("%d", &n);
14     hanoi(n,"A","B","C");
15
16     printf("move %d times\n", time);
17
18     system("pause");
19 }
20
21 void hanoi(int size, char Source, char Temp, char Target)
22 {
23     if (size == 1)
24     {
25         printf("%d step: Move NO.%d , from %c to %c\n", ++time, size, Source,Target);
26     }
27     else
28     {
29         hanoi(size - 1, Source, Target, Temp);
30         printf("%d step: Move NO.%d , from %c to %c\n", ++time, size, Source, Temp);
31         hanoi(size - 1, Source, Target, Temp);
32     }
33 }
```

1. Only one disk can be moved at a time.
2. Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
3. No disk may be placed on top of a smaller disk.



With three disks, the puzzle can be solved in several moves. Above is a program to implement Tower of Hanoi, it shows every steps of Hanoi. Unfortunately, there are some mistakes inside, makes it show wrong steps. Please find the bugs and rewrite it. The format of the answer must be :

(ex) Line XX , return 0 ;

(1) LINE14, hanoi(n,'A','B','C');

(2) LINE30, printf("%d step: Move NO.%d , from %c to %c\n", ++time, size, Source, Target);

(3) LINE 31, hanoi(size - 1, Temp, Source, Target);

(4) If this program works fine, what is the value of move steps (time) when n=3? 7steps

9. Scopes and Durations of Variables

Given the following C code, what is the output?

```
#include <stdio.h>
int i=1000;
void test(void);
int main(void)
{
        int j=77;
        test();
        test();
        test();
        i++;
        printf("In main, i= %d\n",i);
        printf("In main, j= %d\n",j);
}
void test(void)
{
        static int i=99;
        int    j=88;
        printf("In test, i= %d\n",i);
        i++;
        printf("In test, j= %d\n",j);
        j++;
}
```

Ans:

```
        In test, i= 99
        In test, j= 88
        In test, i= 100
        In test, j= 88
        In test, i= 101
        In test, j= 88
        In main, i= 1001
        In main, j= 77
```