**演習問題**（基本的に問題に書かれた動作をすれば正解とします）

①

キーボードから入力された値について以下の処理を行うプログラムを作れ（入力される値は正の整数とする）

a) 1～入力された値までのすべての3の倍数を表示するプログラムを作れ

b) 1～入力された値までの整数の合計を表示せよ

a)ヒント：3の倍数＝3で割った余りが0

②

#include <stdlib.h>を書くことによって rand 関数(戻り値 int 型、引数なし、ランダムな値を返す関数）を使うことができる。

この関数とループを使って乱数を 10 個表示するプログラムを作れ。

※疑似乱数について

この問題の実行結果は毎回同じになるが、それはコンピューターが「種」と呼ばれる値を用いて計算によって乱数を算出しているため。(「疑似乱数」という)

rand 関数はデフォルトの種の値が0になっているのでこの値を変更しないと毎回同じ値が出てくることになる。

乱数の種を変更するには srand 関数（引数に設定したい値を入れる）を使う。

③

以下のような簡単なハイアンドローゲームを作れ

ランダムで 0 から 10 の数字を発生させ、その数字が 5 より大きいか小さいかを当てる。(5 をどちらに含むかは製作者に任せる。)

数字が 5 より大きかったら 1、小さかったら 0 を入力してもらい、それ以外の値が入力されたらプログラムを終了させる。

余裕があればループを用い、0 か 1 以外の値が入力されるまで何回でも遊べるようにしてみること。

ヒント： 0～10 の乱数は rand()%11 と書いてよい。

④(+α)

組み合わせ（コンビネーション）の計算をするプログラムを作れ。ただし、組み合わせの計算には自作関数を作って計算すること。

※組み合わせの計算方法

$$nCr = \frac{n!}{(n-r)!\,r!} = \frac{[n \times (n-1) \times (n-2) \times \cdots \times \{n-(r-1)\}]}{r \times (r-1) \times \cdots \times 2 \times 1}$$

ものすごく余裕があれば再帰関数を使って実装すること。

※再帰関数とは

関数の中で自分自身と同じ関数を呼び出すこと。詳しくは口頭で。

あくまでも解答例です。同じ動作をすればコードが違っていても正解となります。

①a)

```
//3の倍数
#include <stdio.h>
int main(void)
{
	int number;	//入力用
	int i;		//ループ用

	printf("1から入力値までの3の倍数を表示します¥n->");
	scanf("%d", &number);	//入力

	for( i = 1; i <= number; i++)
	{
		if( i % 3 == 0)	//3の倍数＝3で割った余りが0
		{
			printf("%d¥n", i);
		}
	}
	return 0;
}
```

①b)

今回ループをやったのでループを使って解いていますが、$\frac{1}{2}n(n+1)$の公式使って解いた人はそれでも構いません。むしろその発想が大事。

```
//入力値までの合計
#include <stdio.h>
int main(void)
{
	int number;	//入力用
	int sum;	//計算結果
	int i;		//ループ用

	sum = 0;	//あとでこれに足していく
	printf("1から入力値までの合計を算出します。¥n");
	printf("値を入力してください。¥n->");
	scanf("%d", &number);	//入力

	for( i = 1; i <= number; i++)
	{
		sum += i;	//sum = sum + i でもOK
	}
	printf("1から%dまでの合計は%dです。¥n", number, sum);
	return 0;
}
```

②

```c
//乱数を１０個表示。連続で実行しても値が変わらないこと（種固定の疑似乱数であること）も確認しておく。
#include <stdio.h>
#include <stdlib.h>
//#include <time.h> //現在時刻の取得

int main(void)
{
	int num;
	int i;		//ループ用
	//srand( time(0) );  //乱数の種を変える。
	for( i = 0 ; i < 10 ; i++ )
	{
		num = rand();
		printf( "%d¥n" , num );
	}
	return 0;
}
```

③

かなり適当に作ってあります。改良点はたくさんあるので、暇な人はいじってみるといいかも。

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h> //+α

int main(void)
{
	int num;		//乱数格納用
	int hl = 0;		//大きいか小さいか
	srand( time(0) ); //+α

	printf( "値が5より大きいか小さいか当ててください¥n");
	while( hl == 0 || hl == 1 ) //0か1以外が入力されるまでループ
	{
		num = rand()%11; //0～10までの乱数
		printf( "0→5より小さい、1→5より大きい、それ以外→終了¥n->");
		scanf("%d",&hl );
		if(  (hl == 1 && num > 5 ) || ( hl == 0 && num <= 5 ) )
		{
			printf("num = %d ， 正解です¥n",num);
		}
		else
		{
			printf("num = %d ， 不正解です¥n",num);
		}
	}

	printf("終了します¥n");
	return 0;
}
```

④

```c
//再帰関数を使うVer(値が大きすぎると再帰関数の特性でスタックオーバーフロー起こします)
#include <stdio.h>
float combination(int n, int r)
{
	if(r > 0)
		return (n * combination(n-1, r-1) / r);
	else
		return 1;
}

int main()
{
	int n,r;
	printf("Ｎ：");
	scanf("%d",&n);
	printf("Ｒ：");
	scanf("%d",&r);
	printf("Ｎ個中からＲ個取り出す組み合わせ：%f¥n", combination(n, r));
	return 0;
}
```

```c
//再帰関数を使わずにループだけで実装するVer(値が大きすぎると誤差がでます)
#include <stdio.h>
// nCr
float combination(int n, int r)
{
	float result = 1;
	for(int i=0;i<r;i++)
	{
		result *= (float)(n-i)/(r-i);
	}
	return result;
}

int main(void)
{
	int n , r;
	float ans; //結果
	printf( "コンビネーション(nCr)を計算します¥nn かr がマイナスの値なら終了します。¥n" );
	while( 1 )
	{
		printf( "n->" );
		scanf( "%d" , &n );
		printf( "r->" );
		scanf( "%d" , &r );
		if( n < 0 || r < 0 )
			break;

		ans = combination( n , r );
		printf( "combination( %d , %d ) = %f¥n¥n",n,r,ans );

	}
}
```