1. 次の関数 IsPrime() は「引数として与えられた整数が素数ならば true を返し、合成数ならば false を返す」仕様である。空白部分**①②**を以下の選択肢から選び、関数を完成させよ。（配点：２点）

選択肢

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| a. 1 | b. 2 | c. 3 | d. 4 |  |
| e. < | f. <= | g. == | h. = | i. >= |

static bool IsPrime(uint n)

{

if (n < 2) // 0, 1 は素数ではない

{

return false;

}

else if (n == 2) // 2 は素数

{

return true;

}

// 3以上の場合

for (int i = 3 ; i < n; i++)

{

if (n % i == 0) // 割り切れたらそれは素数ではない

{

return false;

}

}

// どれでも割り切れなかったら素数

return true;

}

解答: **①** **②**

1. 以下を埋めて、自然数 n を引数にとり、n 番目のフィボナッチ数を返す関数 Fibonacci() を書け。なお、フィボナッチ数 *Fn* は次の式で定義される数列である。（配点：４点）
   * *F1* =1
   * *F2* =1
   * *Fn* =*Fn-2* +*Fn-1* (*n* ≥ 3)

static uint Fibonacci (uint n)

{

}

1. 以下を埋めて、引数として整数の配列を受け取り、配列の要素となる整数を小さい順（昇順）で並べ替えて返す関数 Sort()を完成させよ。（配点：４点）

static int[]Sort(int[] intArray)

{

for (int i = 0; i < intArray.Length; i++)

{

for (int j = intArray.Length - 1; i < j; j--)

{

if (intArray[j] < intArray[j - 1])

{

int temp = intArray[j];

intArray[j] = intArray[j - 1];

intArray[j - 1] = temp;

}

}

}

return intArray;

}