|  |  |  |  |
| --- | --- | --- | --- |
| 2022/2 『자료구조』실습 보고서 | | | |
| 제목 | 2장 실습(O ) 과제( ) | 제출일자 | 2022.   09.  21 . |
| 학번 | 201911608 | 이름 | 김지환 |

|  |
| --- |
| **실습 2장** |
| 1) Fibonacci 수의 1번째 ~ 20번째 수를 각각 반복과 순환의 실행 결과 비교  #include <stdio.h>  int Fibonacci(int n) {  if (n <= 2) return 1;  return Fibonacci(n - 1) + Fibonacci(n - 2);  }  int main() {  printf(" =========== 순환 ===========");  for (int i = 1; i <= 20;i++) printf("%d번 째 Fibonacci 수 = %d\n", i, Fibonacci(i));  printf(" =========== 반복 ===========");  int fibo[21] = {0,};  fibo[1] = 1;  for (int i = 1; i <= 20; i++) {  if(i>1) fibo[i] = fibo[i - 1] + fibo[i - 2];  printf("%d번 째 Fibonacci 수 = %d\n", i, fibo[i]);  }  return 0;  }  실행결과    -> 실행 결과는 똑같지만  순환의 경우 (n-1) 번과 (n-2) 번을 계속 콜하기 때문에 한 번의 재귀에서 2번의 호출을 하므로 O(2^n)을 갖는다.  반복의 경우 n-1과 n-2만 비교하면 되므로 O(n)의 연산을 가진다.  2) 하노이의 탑 문제, 디스크가 5일 때와 6일 때 이동 결과 출력, 횟수 출력  #include <stdio.h>  int count = 0;  void hanoi\_tower(int n, char from, char tmp, char to) {  count++;  if (n == 1) printf("원판 1, %c -> %c\n", from, to);  else {  hanoi\_tower(n - 1, from, to, tmp);  printf("원판 %d, %c -> %c\n", n, from, to);  hanoi\_tower(n - 1, tmp, from, to);  }  }  int main() {  hanoi\_tower(5, 'A', 'B', 'C');  printf("원판의 총 이동 횟수 : %d\n", count);  count = 0;  hanoi\_tower(6, 'A', 'B', 'C');  printf("원판의 총 이동 횟수 : %d\n", count);  return 0;  }  실행결과    ->  하노이의 탑 총 이동 횟수는 (2^n)-1 이므로  (2^5)-1 = 31회 이다.  6번 실행 결과는 아래에 작성하겠습니다.    ->  (2^6)-1 = 63회 이동 |
| 3) 연습문제 11번  #include <stdio.h>  void asterisk(int i) {  if (i > 1) {  asterisk(i / 2);  asterisk(i / 2);  }  printf("\*");  }  int main() {  asterisk(5);  return 0;  }  실행결과    -> asterisk 가 7번 출력된다. 왜냐하면 i/2를 두번 실행하는 것은 이진트리로 변환되는 구조이다. 즉 5 이런 완전 이진 트리 구조이다. 2^0 + 2^1 + 2^2 = (2^3) - 1 이므로 asterisk(8) 이면  2 2 2^3 이상 2^4 미만 이므로 (2^4)-1인 15개가 출력이 될 것이고 2^0의 경우 1개가 출력  1 1 1 1 될 것이다. 즉, 2^(n-1) 이상, 2^(n) 미만이면 (2^n)-1번 출력된다.  asterisk(0) -  asterisk(2) -  asterisk(4) -  asterisk(8) - |
| 4) 연습문제 12번  #include <stdio.h>  int Sigma(int n) {  if (n == 1) return 1;  return n + Sigma(n - 1);  }  int main() {  printf("\n%d\n", Sigma(10));  return 0;  }  실행결과  -> 1부터 n까지의 합을 순환으로 나타내었을 때 대표적인 1 ~ 10 까지의 합을  출력 해보았습니다. |