1. 상수와 리터럴의 의미와 차이점에 대해 설명하시오.

상수와 리터럴 둘 다 변하지 않는 값을 의미한다.

차이점

상수 : 변하지 않는 변수(클래스나 구조체 같이 기본형에서 파생된 객체나 유도형같은 데이터도 가능)

리터럴 : int I = 1에서 1과 같이 변하지 않는 데이터 (boolean, char, double, long, int... 등)을 리터럴이라 부름

상수와의 차이점으로 리터럴은 클래스 데이터가 올 수 없다.

즉, 상수는 변하지 않는 변수를 의미(메모리 위치) 메모리 값을 변경할 수 없다.

리터럴은 변수의 값이 변하지 않는 데이터(메모리 위치안의 값)를 의미한다.

2. Overloading 과 Overriding 의 각각의 의미와 차이점에 대하여 설명

Overloading : 함수의 이름은 같으나 반환 타입이나 인자의 개수, 인자의 타입이 다른 경우 함수를 재정의 하는 것

ex)

|  |
| --- |
| int sum(int a, int b){  return a + b;  }  int sum(int a, int b, int c){ //<-- 이 경우 overloading  return a + b + c;  } |

Overriding : 상속의 관계에서 부모와 자식의 함수(메소드)가 같은 기능을 수행하지만 함수의 내용이 다른 경우 자식의 클래스에서 함수를 재정의 하는 것.

ex)

|  |
| --- |
| void Parent :: whoAmI(){  printf("Hi~ I'm parent");  }  void Child :: whoAmI(){ //<-- 이 경우 Overriding  printf("Hi~ I'm child);  } |

3. side-effect 란 무엇이며 함수에서 side-effect가 발생하는 경우와 문제점을 각각 설명

정의 : 실행 중에 어떤 객체를 접근해서 변화가 일어나는 행위

꼭 부정적인 것만을 말하는 것은 아니다.

C++은 인수를 평가하는 순서를 정의하지 않아 생기는 경우가 있다.

ex)

|  |
| --- |
| int x = 5;  add(x,++x); |

위의 경우 왼쪽의 인수를 먼저 평가를 하면 5+6 = 11 의결과가 나오고

오른쪽의 인수를 먼저 평가를 하면 6+6 =12 의결과가 나오게 된다.

4. pass-by-reference와 pass-by-value에 대하여 설명하고, 각각 c언어와 c++에서의 경우를 swap()함수를 예시로 설명

pass by reference : 주소에 의한 참조 (원래의 값에 영향)

pass by value : 값에 의한 참조 (원래의 값에 영향이 없음)

|  |
| --- |
| c pass by value |
| #include <stdio.h>  void swap(int a, int b) {  int temp;  temp = a;  a = b;  b = temp;  }  int main() {  int x = 5, y = 3;  printf("swap 함수쓰기전/ x = %d ,y = %d \n", x, y);  swap(x, y);  printf("swap 함수사용후/ x = %d ,y = %d \n", x, y);  return 0;  } |
| 결과  swap 함수쓰기전/ x = 5, y =3  swap 함수사용후/ x = 5, y =3 |

|  |
| --- |
| c pass by reference |
| #include <stdio.h>  void swap(int \*a, int \*b) {  int temp;  temp = \*a;  \*a = \*b;  \*b = temp;  }  int main() {  int x = 5, y = 3;  printf("swap 함수쓰기전/ x = %d ,y = %d \n", x, y);  swap(&x, &y);  printf("swap 함수사용후/ x = %d ,y = %d \n", x, y);  return 0;  } |
| 결과  swap 함수쓰기전/ x = 5, y =3  swap 함수사용후/ x = 3, y =5 |

|  |
| --- |
| c++ pass by value |
| #include <iostream>  using namespace std;  void swap(int a, int b) {  int temp;  temp = a;  a = b;  b = temp;  }  int main() {  int x = 5, y = 3;  printf("swap 함수쓰기전/ x = %d ,y = %d \n", x, y);  swap(x, y);  printf("swap 함수사용후/ x = %d ,y = %d \n", x, y);  return 0;  } |
| 결과  swap 함수쓰기전/ x = 5, y =3  swap 함수사용후/ x = 5, y =3 |

|  |
| --- |
| c++ pass by reference |
| #include <iostream>  using namespace std;  void swap(int &a, int &b) {  int temp;  temp = a  a = b  b = temp;  }  int main() {  int x = 5, y = 3;  printf("swap 함수쓰기전/ x = %d ,y = %d \n", x, y);  swap(x, y);  printf("swap 함수사용후/ x = %d ,y = %d \n", x, y);  return 0;  } |
| 결과  swap 함수쓰기전/ x = 5, y =3  swap 함수사용후/ x = 3, y =5 |

5. 프로그램 실행시간이 0.5초 이상이 되는 연산 및 동작에 대한 예외처리를 디자인 하시오.

(throw - catch, 및 사용자 정의 exception 사용)