1. parameter

a. pass by value
b. pass by reference
c. pass by name
d. pass by result
e. pass by value-result

- side effect 발생하는 것
  b,c,d,e
- aliasing 발생하는 것
  b,c
- java에서 사용하는 것
  a, d
- IN / OUT / INOUT으로 구분
  in - a
  out - d
  inout - b, c, e

3. dynamic chain을 사용하는 이유

1. Stack을 dynamic하게 사용하기 위해
2. 호출부의 stack base를 가리켜 호출부로 돌아갈 때 사용...?

4. static chain을 사용하는 이유

1. nesting 허용 언어에서 non-local 변수에 접근하기 위해

5. thread 생성하는 방법 - 빈칸 채우세요

```
Class MyThread [extends] Thread{
    [public void run(){...}]//필수로 구현해야 하는 것
}
...
Thread myth = new MyThread();
myth.[start()];
```

```
Class MyThread [implements] Runnable{
    [public]//필수로 구현해야 하는 것
}
```

```
...
[Runnable] th = new MyThread();
[Thread] myth = new [Thread](th);
myth.[start()];
```

6. CPU는 [ ]와 [ ]로 구성되어있는데, each instruction is executed in CPU consisting of [ ] and [ ] cycle.

ALU, CU, fetch, decode/execution

7. java에서 예외 관련 키워드 5가지 말하고 설명하세요..

- try
- catch
  예외를 핸들링한다.
  오버로딩 가능하다.
  catch하지 않으면 throws로 호출부에 명시해주어야한다.
- throw
  예외를 발생시킨다.
  java에서는 인스턴스를 throw해야한다.
- throws
  해당 함수에서 에러가 발생할 수 있음을 호출자에게 알린다.
  checked exception을 명시하며, 명시하지 않으면 컴파일 에러가 발생한다.
  catch로 에러 핸들링시 throws 명시하지 않아도 된다.
  c++에서는 없어서 무조건 catch 해야한다. 예외를 전파하는 방식이다.
- final
  catch 구문과 관계없이 무조건 실행되는 문장이다.
  catch 구문 없으면 final 무조건 명시해야한다.

8. java 기준, 예외 발생시키는 방법 두가지

```
throw new MyException();
```

```
MyException e = new MyException();
throw e;
```

9. Activity Record 가 static보다 stack-dynamic 일 때의 장점 2가지

1. 메모리 효율

2. 재귀 가능

10. 예외 발생시킬 때, 메세지를 전달하려면

```
Class MyExcetption extends Exception{
    public MyException(String message){
        [f.addWindowListener(new EventHandler());]
    }
}
```

11. Overloaded Subprograms을 구현하는 것 (2가지)

template - c++
generic - java

13. Loop 종류 3가지 및 사용 예시

counter - for
iterator - data structure
logic - do-while, while

14. Thread 설명 중 옳은 것?

a. It is a scheduling unit
b. Lightweight Process
c. 프로세스 간 커뮤니케이션에 사용된다.
d. run 메소드는 스레드의 생성자이다.
e. t1.join()은 스레드 종료에 사용된다.
f. 여러 생성자를 가질 수 있다.
g. 하나의 클래스에서 여러 스레드를 가질 수 있다.

a, b, c, f, g

15. cooperation, competition, race condition에 대해 설명하세요

- cooperation
  차례대로 기다려서 공유자원에 접근한다. wait, notify, notifyAll
  parallel, pipeline이 존재하며 producer - consumer관계가 있다.
  앞의 output이 뒤의 input으로 된다.

- competition
  공유자원에 경쟁적으로 접근한다.
  lock/unlock을 통해 구현하며 상호배타성을 가진다.

race condition이 발생한다.

- race condition
  공유자원에 두개 이상의 스레드/프로세스가 동시에 접근할 때 발생한다.

16. Activation Record Instance에 포함되는 값을 순서대로 나열하고 설명하세용.

top

- local var
- [argument] // 파라미터 존재시
- dynamic link // 호출부의 stackbase
- [static link] //자신의 부모
- return pc

17. Exception Propagation이란?

예외가 처리(catch)될 때까지 호출부로 control이 넘어가는 것을 말한다.
throws로 구현되며, main까지 처리되지 않는다면 abort

18. checked Exception과 unchecked Exception에 대해 설명하고 사용자 정의 unchecked 예외를 만드는 방법?

- checked
  예외 발생 가능성을 미리 알 수 있다.
- unchecked
  컴파일러도 예외가 발생할 지 체크할 수 없다. catch할 수 있지만 권장되지 않는다.
- 사용자 정의 unchecked 예외

```
class MyEx extends RuntimeException {}
```

19. Event Handling과 Exception Handling의 차이점 한가지

event - 이벤트 발생시 처리하지 않아도 아무 일이 일어나지 않는다.
exception - 예외 발생시 처리하지 않으면 컴파일 에러나 abort가 발생한다.

20. Event Handling Code

```
class FrameTest3{
    public static void main(String args[]){
        Frame f = new Frame("Login");
        f.setSize(300,200);

        f.addWindowListener(new EventHandler())
```

```
            f.setVisible(true);
        }
    }
```

21. Class에 관련된 연산 4가지

constructor, destructor, iterator, accessor

22. function과 procedure의 차이점

function - 명시적인 리턴값 존재
prcedure - 리턴값이 존재하지 않는다.
전역 변수같은 자신 외의 변수를 변경하는 방식이라 side effect, aliasing이 발생할 수 있다.

23. JAVA GUI component에서 발생하는 걸 뭐라고 하나?

event handling

24. positional parameter와 keyword parameter의 정의 및 keyword parameter의 장단점

- positional
  actual, formal parameter 값의 전달 기준이 위치로 결정된다.
- keyword
  actual parameter에서 formal parameter에 전달할 값을 formal parameter의 변수와 함께 명시적으로 지정할 수 있다.
  - 장점: 변수 전달시 위치를 고려하지 않아도 된다.
  - 단점: formal parameter의 이름을 알아야한다.