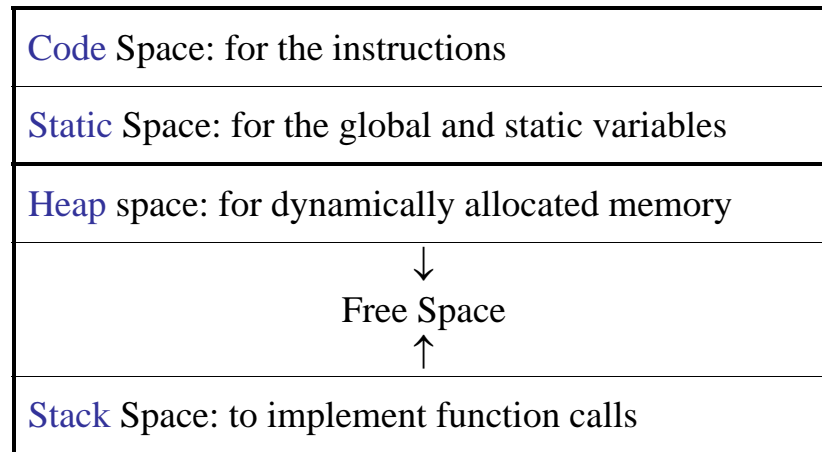


STORAGE ORGANISATION

- When a program is loaded into memory, the OS will give it memory to work with. This memory consists of:



- Code and Static space are static: The compiler generates precise requirements for code and static space.
- Heap and Stack space are dynamic:
 - The compiler can request approximate requirements for dynamic space.
 - The OS manages additional requests to expand dynamic space.

FUNCTION CALLS

- Function calls and returns are managed using a run-time stack called the **control stack** stored in the stack space.
- Each activation of a function has an **activation record** or **stack frame** on the stack.
- Activation record:

<i>Access links: links to data found in other activation records</i>
<i>Space for unnamed local data</i>
Local variables
Parameter values
<i>Saved machine status: usually registers</i>
Return address: location counter
Return value

- Example: Fibonacci function

```
int Fib(int n) {
    int n1, n2;
    if (n<=0)    { /* error */}
    if (n<=2)    return 1;
    n1 = Fib(n-1); // line 5
    n2 = Fib(n-2); // line 6
    return n1+n2;
}
Fib(4);          // line 9
```

- **Calling sequence**: code that allocates an activation record on stack and enters information into its fields.
- **Return sequence**: code that restores state of machine so calling procedure can continue.