Macchina astratta

Dato un linguaggio di programmazione L, una macchina astratta per L è un insieme M_L di algoritmi e strutture dati in grado di *memorizzare* ed *eseguire* programmi in L.
Le macchine astratte sono caratterizzate dai seguenti componenti:

memoria: conserva dati e programmi;

operazioni primitive: istruzioni pensate per operare efficientemente sulle risorse a disposizione della macchina astratta, tenendo in considerazione le caratteristiche proprie del linguaggio.

interprete: esegue le istruzioni contenute nei programmi, traducendoli in operazioni primitive e operando sulla memoria;

controllo: algoritmi e strutture dati usati dall'interprete per gestire flusso di esecuzione, acquisire la prossima istruzione e i suoi operandi, chiamate di funzioni, thread, ambiente, accesso e gestione della memoria...

Esempi

Elementi caratteristici delle macchine astratte di qualche linguaggio:

Algol: stack, heap, program store. Primitive: accesso a variabili e array, scope, call-by-value, call-by-name (simile a sostituzione dei parametri nel corpo della funzione).

Smalltalk: macchina stack-based (no registri) con relative primitive, più istruzioni per la gestione di oggetti (metodi, variabili di istanza...).

Python: stack-based con molte primitive.

Linguaggi funzionali: stack per le chiamate, ambiente, chiusure per rappresentare il valore di una funzione, heap con GC.

Macchine fisiche: memoria hardware e registri, assembly, ALU (interprete), fetch-execute e Program Counter (controllo).