

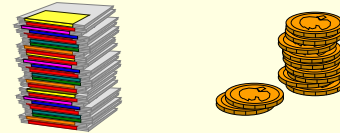
Pilha

SCC0502 – Algoritmos e Estruturas de Dados I

Prof. Thiago A. S. Pardo

Pilha

- O que é?
- Para que serve?



2

Problema: chamada de sub-rotinas

Rotina A	Rotina B	Rotina C	Rotina D
1 print "A"	1 call C	1 print "C"	1 print "D"
2 call C	2 print "B"	2 call D	2 return
3 call B	3 call D	3 return	
4 call D	4 call C		
5 return	5 return		

Qual o resultado da execução da rotina A?

3

Problema: chamada de sub-rotinas

Rotina A	Rotina B	Rotina C	Rotina D
1 print "A"	1 call C	1 print "C"	1 print "D"
2 call C	2 print "B"	2 call D	2 return
3 call B	3 call D	3 return	
4 call D	4 call C		
5 return	5 return		

Qual o resultado da execução da rotina A?

Qual a dificuldade para se fazer esse cálculo?

4

Problema: chamada de sub-rotinas

Rotina A	Rotina B	Rotina C	Rotina D
1 print "A"	1 call C	1 print "C"	1 print "D"
2 call C	2 print "B"	2 call D	2 return
3 call B	3 call D	3 return	
4 call D	4 call C		
5 return	5 return		

Qual o resultado da execução da rotina A?

Qual a dificuldade para se fazer esse cálculo?

Possíveis soluções?

5

Problema: chamada de sub-rotinas

1. Um computador está executando a rotina X e, **durante a execução de X**, encontra uma **chamada à rotina Y**
2. **Para-se a execução de X** e se **inicia a execução de Y**
3. Quando se **termina a execução de Y**, o computador deve saber o que fazer, isto é, **onde voltar na rotina X**

6

Problema: chamada de sub-rotinas

■ Dificuldade

- O que estava sendo executado quando uma sub-rotina foi interrompida? Para onde voltar agora que se chegou ao fim de uma sub-rotina?

■ Solução

- A cada chamada de sub-rotina, armazenar o endereço de retorno (rotina e número da linha, por exemplo)
- Como armazenar o endereço de retorno de chamadas sucessivas: pilha

7

Pilha (*stack*)

■ Definição

- Estrutura para armazenar um conjunto de elementos que funciona da seguinte forma
 - Novos elementos sempre entram no "topo" da pilha
 - O único elemento que se pode retirar da pilha em um dado momento é o elemento do topo

■ Para que serve

- Para modelar situações em que é preciso "guardar para mais tarde" vários elementos e "lembrar" sempre do último elemento armazenado

■ L.I.F.O.

- Last In, First Out

8

Problema: chamada de sub-rotinas

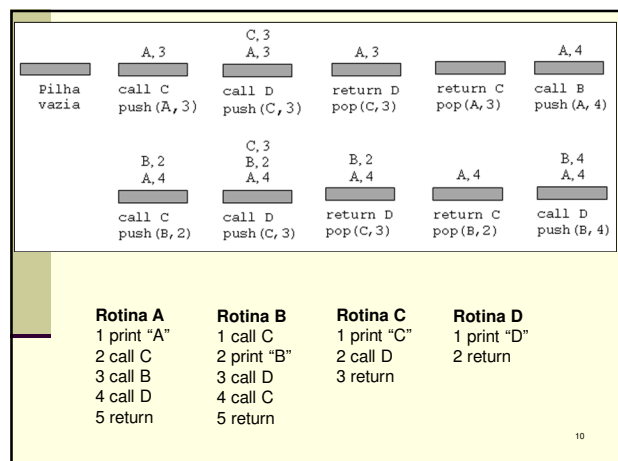
■ A cada comando **call**

- Empilha (*push*) o endereço para retornar depois
- Passa a executar a nova sub-rotina

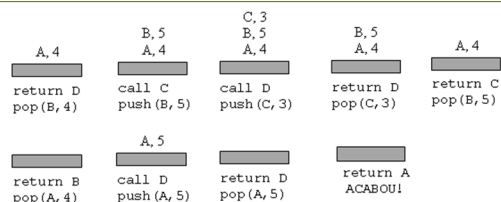
■ A cada comando **return**

- Desempilha (*pop*) o último endereço armazenado
- Passa a executar a partir do endereço desempilhado

9



10



Rotina A	Rotina B	Rotina C	Rotina D
1 print "A"	1 call C	1 print "C"	1 print "D"
2 call C	2 print "B"	2 call D	2 return
3 call B	3 call D	3 return	
4 call D	4 call C		
5 return	5 return		

11

Problema: chamada de sub-rotinas

■ Resultado

- A, C, D, C, D, B, D, C, D, D

12

Pilha

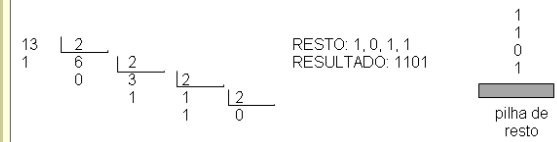
Operações usuais

- **Push(P,X)**: empilha o valor da variável X na pilha P
- **Pop(P,X)**: desempilha P e retorna em X o valor do elemento que estava no topo de P
- **X=top(P)**: acessa o valor do elemento do topo de P, sem desempilhar
- **Create(P)**: cria uma pilha vazia P
- **Y=IsEmpty(P)**: Y recebe *true* se a pilha estiver vazia; *false* caso contrário
- **Empty(P)**: esvazia uma pilha P

13

Exercício: outra aplicação

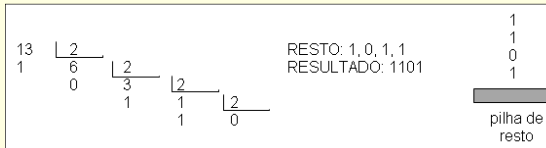
Fazer algoritmo de conversão decimal para binário usando pilha



14

Exercício: outra aplicação

Fazer algoritmo de conversão decimal para binário usando pilha



Estratégia de resolução:

- a cada divisão, empilha o resto
- quando acabar a divisão (quociente=0), desempilha e escreve todos os elementos

Considerar prontos: operações da pilha, resto(X,Y) e quociente(X,Y)

15

Exercício: outra aplicação

Variáveis

- P: pilha
- N: inteiro {número a ser convertido}
- X: inteiro {resto da divisão}

Início do algoritmo

```
leia N
create(P)
repta
  X=resto(N,2)
  push(P,X)
  N=quociente(N,2)
até que (N=0)
  escreva "o resultado é "
  enquanto (IsEmpty(P)=falso) faça
    pop(P,X)
    escreva X
```

Fim

16

Implementação da pilha

Alocação sequencial

- Os elementos da pilha ficam, necessariamente, em sequência (um ao lado do outro) na memória

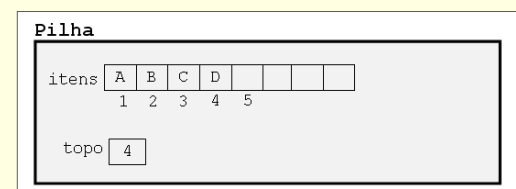
Alocação estática

- Todo o espaço de memória a ser utilizado pela pilha é reservado (alocado) em tempo de compilação
- Todo o espaço reservado permanece reservado durante todo o tempo de execução do programa, independentemente de estar sendo efetivamente usado ou não

17

Implementação da pilha

Sequencial e estática



18

Implementação da pilha

■ Declaração em C

?

19

Implementação da pilha

■ Declaração em C

```
#define TamPilha 100  
  
typedef int elem;  
  
typedef struct {  
    int topo;  
    elem itens[TamPilha];  
} Pilha;  
  
Pilha P;
```

20

Exercício

■ Implementar operações da pilha

- Create
- Empty
- IsEmpty
- IsFull
- Push
- Pop
- Top

■ Atenção: considerações sobre TAD

- Arquivos .c e .h, parâmetros, mensagens de erro

21