Lista de Exercicios

Aluno: Paulo Fabio

1º) Linguagem de alto nivel: Linguagens que possuem uma estrutura que mais se assemelham a escrita humana, o que permite que pessoas que nunca tiveram contato com uma linguagem de programação entenda um pouco de como ela funciona.
Linguagem de baixo nivel: Linguagens que funcionam a nivel de hardware, ou seja, são as linguagens que trabalham diretamente com instruçoes do processador para realizar suas funçoes .
O **compilador** é responsavel por pegar um codigo gerado em uma linguagem de alto nivel e traduzir para uma linguagem que o computador entenda, ou seja, ele gera um codigo na linguagem de maquina equivalente ao que foi desenvolvido pelo programador.

2º) Em C:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main()
{
 int i,n;
 float soma=1;
 scanf("%d", &n);

 for(i=1;i<=n;i++){
    if(i%2)
      soma=soma*(float)pow(2*i-1,2*i-1)/pow(2*i,2*i);
    else
      soma=soma+(float)pow(2*i-1,2*i-1)/pow(2*i,2*i);
 }
 printf("%f\n",soma);
 return 0;
}
```

Em C++:

```
#include <iostream>
#include <math.h>
using namespace std;
int main(){
            int i,n;
            float soma=1;
             cin >> n;
             for(i=1;i<=n;i++){
                        if(i%2)
                         soma=soma*(float)pow(2*i-1,2*i-1)/pow(2*i,2*i);
```

```
                    else
                    soma=soma+(float)pow(2*i-1,2*i-1)/pow(2*i,2*i);}
                    cout << soma << endl;
 return 0;
}
```

Em python:

```
n=int(input("entre com o valor de n: "))
soma=1
for i in range(1,(n+1)):
if(i%2==1):
soma=soma*(((2*i)-1)**((2*i)-1))/((2*i)**(2*i))
else:
soma=soma+(((2*i)-1)**((2*i)-1))/((2*i)**(2*i))
print(soma)
```

Em perl:

```
#!/usr/bin/perl

print "Digite o N desejado:\n";
my $n = <>;
my $res = 1.0;
my $i = 1;

for($i=1;$i<=$n;$i++){
        if($i%2 == 1){
                $res = $res*(((2*$i)-1)**((2*$i)-1))/((2*$i)**(2*$i));
        } else {
                $res = $res+(((2*$i)-1)**((2*$i)-1))/((2*$i)**(2*$i));
        }

}
printf "%.15f\n",$res;
```

3º)

Em C:

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int main() {
char nomes[10][15], aux[15];
int quantidade, i, j;
printf("Quantidade total de nomes: ");
scanf("%d", &quantidade);
```

```
for(i=0; i<quantidade; i++)
        scanf("%s", nomes[i]);
for(i=0; i<quantidade; i++) {
                for(j=0; j<quantidade; j++) {
                        if(strcmp(nomes[i], nomes[j]) < 0) {
                                strcpy(aux, nomes[i]);
                                strcpy(nomes[i], nomes[j]);
                                strcpy(nomes[j], aux);
                        }
                }
        }
printf("\n-Nomes Selecionados:-\n");
for(i=0; i<2; i++)
printf("%s\n", nomes[i]);
return 0;
}
```

Em C++:

```
#include<stdio.h>
#include<string.h>
#include <iostream>
#include <cstdlib>
using namespace std;
int main() {
char nomes[10][15], aux[15];
int quantidade, i, j;
cout << "Quantidade total de nomes: ";
cin >> quantidade;
for(i=0; i<quantidade; i++)
   cin >> nomes[i];
   for(i=0; i<quantidade; i++) {
      for(j=0; j<quantidade; j++) {
         if(strcmp(nomes[i], nomes[j]) < 0) {
            strcpy(aux, nomes[i]);
            strcpy(nomes[i], nomes[j]);
            strcpy(nomes[j], aux);
         }
      }
   }
cout << "\n--Nomes Selecionados:--";
for(i=0; i<2; i++)
   cout << "\n" << nomes[i];
return 0;
}
```

Em Python:

```
i=0
tam=int(input("Digite a quantidade de nomes: "))
nomes=[]
for i in range(tam):
        nome=str(input())
        nomes.append(nome)
nomes.sort()
print ("\n")
print (nomes[0])
print (nomes[1])
```

4º)

5º)a) Imperativas: As linguagens imperativas são orientadas a ações, onde a computação é vista como uma sequência de instruções que manipulam valores de variáveis. Ex: Java, Python.
b) Funcionais : São linguagens que enfatizam o uso de funções matematicas para se chegar a uma saida do programa. Ex: Haskell, Lisp.
c) Lógicas : Programação lógica é um tipo de programação a qual utiliza uso de lógica matemática em sua essência. Ex: PROLOG
d) Marcação/Híbrida: É representada por um conjunto de códigos empregados a dados e a textos, com a finalidade de adicionar informações específicas sobre esse dado ou texto. Além disso são linguagens que também oferecem suporte para o desenvolvimento de programas funcionais.

6) <expr> → <expo> * <expr> | <expo>
<expo>→ <base> ^ <expo> | <base>
<base>→ A | B | C

7) É um modelo, padrão ou estilo de programação suportado por linguagens que agrupam certas características comuns, fazendo com que as linguagens de programações existentes possam ser agrupadas de acordo com seu paradigma, possibilitando que um programador que está em busca de aprender uma nova linguagem de programação ja tenha uma noção de como ela ira se comportar.

8) Analisador lexico: É responsavel por fazer uma varredura no programa, lendo caractere por caractere e montando uma sequencia de tokens que serao usados no analisador sintatico, o analisador lexico também é responsavel por reconhecer as palavras reservadas, constantes e outras palavaras que pertecem a linguagem.

9.1) imagem 01.

9.2) A analise semantica tem por função a verificação de tipos, a verificação do fluxo de controle e a verificação da unicidade da declaração de variáveis. A análise semântica trata a entrada sintática e transforma-a numa representação mais simples e mais adaptada a geração de código. Esta camada

do compilador fica igualmente encarregada de analisar a utilização dos identificadores e de ligar cada uma delas a sua declaração.
Semantica operacional: Descreve o significado de um programa pela execução de suas instruções em uma máquina, seja ela real ou simulada.
Semântica axiomática: baseia-se na lógica matemática associada a um método para provar, ou seja, preocupa-se em provar a exatidão de um programa. As expressões lógicas são chamadas de predicado. Eles precedem uma instrução, seguido da instrução temos a pós-condição.
Semântica denotacional: dentre os métodos mais empregados, é o de maior rigor matemático. Baseia-se solidamente na teoria das funções recursivas, ou seja, explica o significado das instruções das linguagens, através de instruções matemáticas.

10) a) 122*y -144>144
122*y>144+144
122*y>288
y>288 / 122

b) y=5*x-5
y+5<45
y<40

5*x-5<40
5*x
x<9

c) y>2
y+2>2
y>0
y=y-2
y-2>2
y>4

d) zero iterações: {i=N}
uma iteração: wp=(i = i + 1 , i = N) ={ i + 1 = N} ou { i = N – 1}
duas iterações wp=(i = i + 1 , i = N -1) ={ i + 1 = N -1} ou { i = N – 2}

logo: precondição { i <=N}