

**MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
CATARINENSE CAMPUS VIDEIRA**

**Curso de Educação Profissional Técnica de Nível Médio
Integrado em Informática**

Relatório de Estágio Curricular

Aluno

**Videira – SC
Novembro, 2019**

**MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
CATARINENSE CAMPUS VIDEIRA**

Aluno

Relatório de Estágio Curricular

Relatório de Estágio Curricular
Obrigatório apresentado como parte
das atividades para obtenção do título
de Técnico em Informática, do
CEPTNM integrado, do Instituto
Federal de Educação, Ciência e
Tecnologia Catarinense – Câmpus
Videira, SC.

Aprovado por:

Kennedy Araújo
(Orientador)

Maurício Natanael
Ferreira

Gerson Luiz Camillo

**Videira – SC
Novembro, 2019**

Sumário:

IDENTIFICAÇÃO	5
INTRODUÇÃO	6
APRESENTAÇÃO DO INSTITUTO	7
ATIVIDADES REALIZADAS	8
ESTUDO DA LINGUAGEM PYTHON	8
CRIAÇÃO DE CARDS EXPLICATIVOS	10
CRIAÇÃO DE LISTAS DE EXERCÍCIOS	15
APRESENTAÇÃO DE FERRAMENTAS PARA OBI	17
RESOLUÇÃO COMENTADA DAS QUESTÕES	19
CATEGORIZAÇÃO DAS QUESTÕES DA OBI	21
CONCLUSÃO	23
REFERÊNCIAS	24

Lista de Ilustrações:

Figura 1 - Foto de um laboratório de Informática	7
Figura 2 - Código em Python	9
Figura 3 - Card sobre Comandos de Entrada e Saída e Operadores	11
Figura 4 - Card sobre IF	12
Figura 5 - Card sobre FOR	13
Figura 6 - Listas de exercícios compartilhadas	105
Figura 7 - Slide 2 da Apresentação	157
Figura 8 - Exemplo de resolução de lista	179
Figura 9 - Exemplo de questão da OBI	192

IDENTIFICAÇÃO**IDENTIFICAÇÃO DO INSTITUTO****Instituto Federal Catarinense****Endereço:** Rodovia SC 135, KM 125**Bairro:** Campo Experimental**Cidade:** Videira**UF:** SC**CEP:** 89564-550**Número Telefone:** (49) 3533-4900**Email:** ifc@ifc-videira.edu.br**IDENTIFICAÇÃO DO ESTAGIÁRIO****Leonardo Martins****Período:** 3º Ano**Turma:** 2017**Matrícula:** 2017319278**Email:****IDENTIFICAÇÃO DO ESTÁGIO****Aprendizado sobre Python****Data de Início:****Data de Fim:****Carga Horária Total:****Média de Horas Semanal:****Supervisor:****Email:** kennedy.araujo@ifc.edu.br**IDENTIFICAÇÃO DE ATIVIDADES**

- 1** Estudo da Linguagem
- 2** Elaboração de cards explicativos
- 3** Criação de listas de questões
- 4** Apresentação da Linguagem Python para alunos
- 5** Resolução comentada das questões
- 6** Categorização de questões da OBI

1. INTRODUÇÃO

Com o objetivo de formação na grade técnica do curso, é necessário que sejam realizadas atividades técnicas na área sendo elas através do estágio, que possibilita ao estudante colocar em prática seus conhecimentos e adquirir novos com as situações encontradas no ambiente real de trabalho, com auxílio de um profissional experiente, além de ser um requisito obrigatório do Projeto Pedagógico de Curso (PPC), do Curso de Educação Profissional Técnica de Nível Médio (CEPTNM) Integrado em Informática do Instituto Federal Catarinense – Campus Videira.

Sendo que a montagem e organização do relatório de estágio é de suma importância para a aprendizagem e formação do estudante, pois é nele que são organizadas as atividades práticas realizadas e suas descrições e métodos de como fazer tais tarefas, a forma de como escrever o relatório deve ser bem clara e objetiva demonstrando suas práticas e habilidades no período de estágio.

Nesse relatório serão apresentadas atividades no período de 18 de fevereiro de 2019 até o dia 31 de julho de 2019, realizado no próprio Instituto Federal Catarinense com orientação do professor Kennedy Araújo. As atividades estão descritas com dificuldade, objetivo e quais foram as experiências adquiridas na realização das tarefas.

O relatório está organizado em cinco seções, sendo esta, a primeira seção, onde apresenta-se o trabalho, na segunda é feito uma breve exposição da instituição em que o estágio foi realizado, já a terceira descreve as atividades realizadas e é dividida em seis subseções detalhando cada atividade, a quarta seção contém as considerações finais e, finalmente, na quinta e última seção apresentam-se todas as referências bibliográficas utilizadas no decorrer do trabalho.

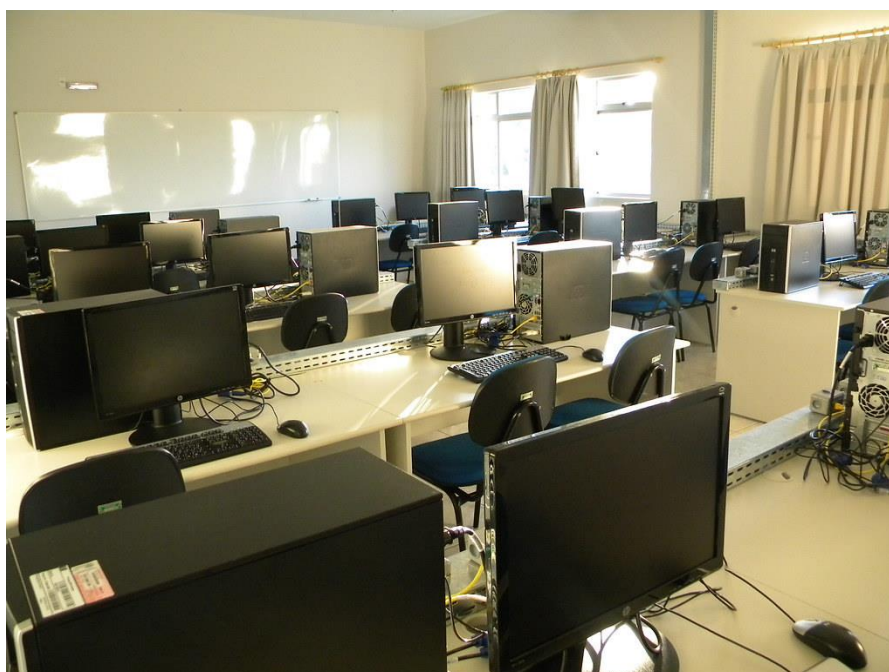
2. APRESENTAÇÃO DO INSTITUTO

O Instituto Federal Catarinense (IFC) possui atualmente 15 Campus, segundo o site do mesmo, distribuídos pelo estado, além da Reitoria que está instalada na cidade de Blumenau.

O IFC oferece educação a partir do nível médio, até a graduação. Preferencialmente, buscamos o atendimento das demandas regionais de localização dos campus, com isso esperamos a interferência positiva na transformação da realidade social e econômica, contribuindo para o desenvolvimento dos arranjos produtivos locais e regionais.

A instituição atua em diversas áreas com cursos técnicos em agropecuária, informática e eletroeletrônica. Ainda, há cursos de nível superior nas áreas de ciência da computação, engenharia elétrica e pedagogia. Além disso, o campus possui programas de bolsa de Pesquisa e Iniciação Científica, de Extensão que contemplam importantes atividades nos câmpus, despertando nos estudantes a curiosidade e o interesse em buscar e desenvolver conhecimento além das atividades rotineiras.

Figura 1 - Foto de um laboratório de informática.



Fonte: Instituto Federal Catarinense

3. ATIVIDADES REALIZADAS

Nesta lista segue todas as atividades realizadas no estágio e suas respectivas informações.

3.1. ESTUDO DA LINGUAGEM PYTHON

Tempo Estimado		Horas	Atividade
Início	Final		
18/02/19	10/07/19	30	Estudo sobre Python

Objetivo: Aprender a sintaxe e lógica da programação de Python.

Porque foi feito: Para que os estudantes obtivessem conhecimento condizentes com o estágio também se tivessem preparo para a prova da OBI.

Descrição: O aprendizado foi construído com ajuda do curso Python para Zumbis¹, indicado pelo professor Kennedy, e com outros vídeos como a série do curso em vídeo de Gustavo Guanabara². Esses vídeos ensinavam de forma detalhada e simples como programar em Python desde de seu início, sendo assim fácil de assimilar o que foi aprendido e botar em prática nos exercícios. A sequência de vídeos começa especificando a sintaxe e o básico da linguagem e depois vai aprofundando em estruturas com IF-ELSE, FOR, vetores e etc.

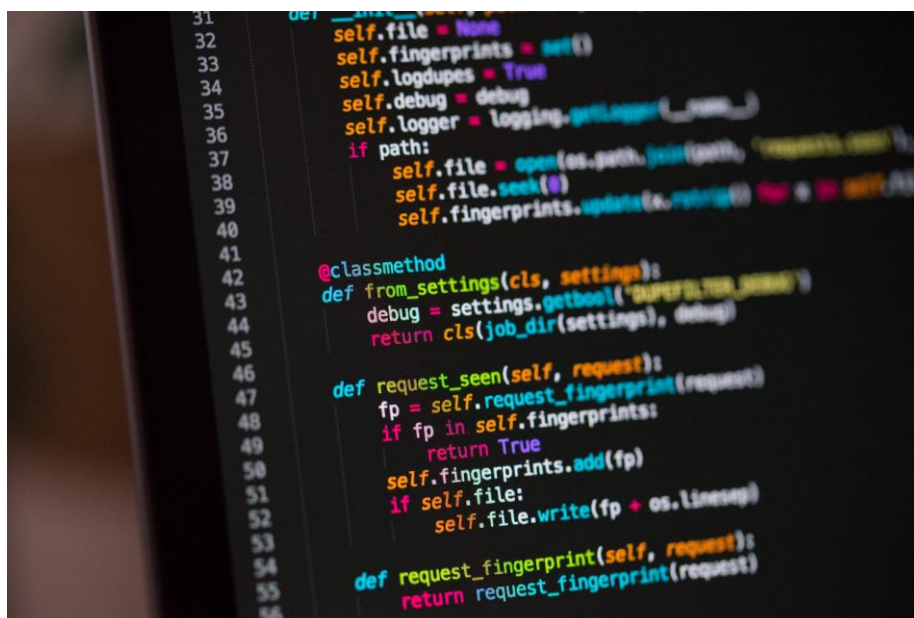
Durante essa etapa pudemos observar que linguagem tem uma estrutura sintática diferentes daquele conhecida. Em Python, os blocos de código são delimitados pelo uso de endentação, que deve ser constante no bloco de código, mas é considerado um bom padrão manter a consistência no projeto todo e evitar a mistura tabulações e espaços. A linha anterior ao bloco sempre termina com dois pontos e representa uma estrutura de controle da linguagem ou uma declaração de uma nova estrutura (uma função, por exemplo). (BORGES, 2010).

¹ <https://www.pycursos.com/python-para-zumbis/>

² <https://www.cursoemvideo.com/course/curso-python-3/>

Alguns sites também auxiliaram no aprendizado como o eXcript³, DevMedia⁴ e W3School⁵, onde se pode encontrar definições, exemplos e dúvidas sobre a linguagem Python, assim como várias outras. Abaixo um exemplo de codificação nesta linguagem.

Figura 2 - Código em Python.



```
31 def __init__(self):
32     self.file = None
33     self.fingerprints = set()
34     self.logdupes = True
35     self.debug = debug
36     self.logger = logging.getLogger(__name__)
37     if path:
38         self.file = open(os.path.join(path, 'requests.txt'),
39                         'a')
40         self.file.seek(0)
41         self.fingerprints.update([x.strip() for x in self.file.readlines()])
42
43 @classmethod
44 def from_settings(cls, settings):
45     debug = settings.getbool('SUPERFINGER_PRINT')
46     return cls(job_dir(settings), debug)
47
48 def request_seen(self, request):
49     fp = self.request_fingerprint(request)
50     if fp in self.fingerprints:
51         return True
52     self.fingerprints.add(fp)
53     if self.file:
54         self.file.write(fp + os.linesep)
55
56 def request_fingerprint(self, request):
57     return request_fingerprint(request)
```

Fonte: Chris Ried/Unsplashd.

Dificuldades: Nenhuma dificuldade apresentada.

Qual foi a aprendizagem: A base de uma linguagem inteiramente nova, na qual pudemos aplicar o conhecimento que já tínhamos dos anos anteriores. Com esse conteúdo aprendemos as principais estruturas e quando e onde utilizá-las, de modo que tudo aprendido foi útil.

³ <http://excript.com/curso-de-python.html>

⁴ <https://www.devmedia.com.br/guia/python/37024>

⁵ <https://www.w3schools.com/python/>

3.2. CRIAÇÃO DE CARDS EXPLICATIVOS

Tempo Estimado		Horas	Atividade
Início	Final		
25/04/19	04/07/19	10	Criação de <i>Cards</i>

Objetivo: Produzir material para a consulta dos futuros alunos que participarão do projeto da OBI numa linguagem direta e simples.

Por que foi feito: Os *cards* explicativos foram feitos para auxiliar no processo de aprendizagem da linguagem Python, possibilitando aos futuros integrantes do projeto passar por essa etapa de maneira mais ágil.

Descrição: Primeiramente aprendemos o máximo sobre a linguagem para que pudéssemos fazer os *cards* da melhor forma possível e também para que estes não contivessem erros.

Para a atividade, foram separados em 8 tópicos subdividindo sobre qual assunto cada card trataria. São estes:

- Linguagem Python;
- Comandos de entrada e saída e operadores aritméticos;
- Condicional *IF-ELIF-ELSE*;
- Estrutura de Repetição *FOR*;
- Estrutura de Loop *WHILE*;
- Matriz e vetor;
- Funções;
- Funções para o tratamento de *Strings*;

Cada aluno ficou responsável por três destes temas. O primeiro *card* abordava os “comandos de entrada e saída e operadores aritméticos” onde se mostra como se deve fazer a entrada de dados com o código *Input* e a saída dos mesmos com o *Print*. Além de mostrar como se deve fazer operações como potenciação ou divisão.

Figura 3 - Card sobre Comandos de Entrada e Saída e Operadores.

ENTRADA/SAÍDA & OPERADORES ARITMÉTICOS

OBI2019 - Leonardo Martins

ENTRADA

Refere-se a entrada de dados do mundo externo para o meio digital. É dessa forma que enviamos informações para dentro das nossas aplicações.

Exemplo: `input("Digite a informação")`

SAÍDA

É a saída de dados do código para o console ou outro lugar. De forma que, sirva para informar ou repassar algo útil ao usuário.

Exemplo: `print("Essa é a saída de dados")`

OPERADORES ARITMÉTICOS

O Python pode ser utilizado como uma calculadora matemática avançada. Praticamente, todos os operadores aritméticos funcionam da mesma forma como os conhecemos da matemática elementar.

SOMA

Exemplo: `soma = 8 + 8`

SUBTRAÇÃO

Exemplo: `sub = 8 - 8`

MULTIPLICAÇÃO

Exemplo: `multi = 8 * 8`

EXPONENCIAÇÃO

Exemplo: `div = 8 / 8`

PARTE INTEIRA

Exemplo: `div = 8 // 8`

MÓDULO

Exemplo: `div = 8 % 8`

Fonte: O Autor

O segundo *card* foi sobre “Condicional - IF”, no qual foi explicado como funciona as cláusulas *if*(se), *elif*(senão) e *elif*(senão se), dentro das quais é necessário uma condição que compare (com maior, menor, igual) dois valores e

retorne verdadeiro ou falso para que o bloco de código abaixo seja executado ou não na sintaxe do Python.

Figura 4 - Card sobre IF.

OBI - 2019: Leonardo Martins

Condicional IF

O **if** é uma estrutura de condição que permite avaliar uma expressão e, de acordo com seu resultado, executar uma determinada ação.

```
if(condição==true):           if(a > b):
    execute()                  print("A é maior que B")
```

A condição deve estar entre parênteses e com algum sinal de comparação, como: **<**, **>**, **>=**, **<=**, **==** ou **!=**

Quando for necessário duas condições deve se usar o comando **AND** ou **OR** entre as condições

AND: É preciso que as duas condições sejam verdadeira para que o código seja executado.

OR: É preciso que apenas uma das condições seja verdadeira para que o código funcione.

Quando é necessário um comportamento específico para definir caso de a condição não ser executada, precisamos utilizar a reservada **else**.

```
idade = 18
if idade >= 18:
    print('maior de idade')
else:
    print('menor de idade')
```

Se existir mais de uma condição alternativa que precisa ser verificada, devemos utilizar o **elif** antes de usar o **else**, da seguinte forma:

```
if idade < 12:
    print('criança')
elif idade < 18:
    print('adolescente')
elif idade < 60:
    print('adulto')
else:
    print('idoso')
```

Fonte: O Autor

Por fim, o último *card* foi sobre “Estrutura de repetição **FOR**” onde um laço de repetição que irá repetir a linha de código até que a condição seja completada. O laço **FOR** nos permite percorrer os itens de uma coleção e, para cada um deles, executar o bloco de código declarado no *loop*.

Figura 5 - Card sobre FOR



```

FOR N IN RANGE(10):
    PRINT(N)
    //CÓDIGO PARA IMPRIMIR
    //VARIÁVEL N 10 VEZES
  
```

A ESTRUTURA **FOR** EXIGE, INICIALMENTE, A DEFINIÇÃO DE UMA VARIÁVEL E, EM SEGUIDA, A LISTA QUE SERÁ ITERADA. A SEGUIR, TEMOS O ESQUEMA PARA O USO DA INSTRUÇÃO **FOR**.

```

FOR <VARIÁVEL> IN <OBJETO ITERÁVEL>:
    ///CÓDIGO
  
```

A ESTRUTURA DE REPETIÇÃO **FOR** EXECUTA UM CICLO PARA CADA ELEMENTO DO OBJETO QUE ESTÁ SENDO ITERADO. A FORMA MAIS SIMPLES, É GERANDO UMA LISTA COM A FUNÇÃO **RANGE()**.

É POSSÍVEL ADICIONAR A INSTRUÇÃO **ELSE** AO FINAL DO **FOR**. ISSO FAZ COM QUE UM BLOCO DE CÓDIGO SEJA EXECUTADO AO FINAL DA ITERAÇÃO

```

FOR N IN RANGE(X):
    //REPETIÇÃO X VEZES
ELSE:
    //CÓDIGO PARA SER EXECUTADO DEPOIS
  
```

ASSIM COMO O **FOR** PODE RODAR UMA LISTA INTEIRA:

```

NOMES = ['PEDRO', 'JOÃO', 'LETICIA']
FOR N IN NOMES:
    PRINT(N)
  
```

```

>>>
PEDRO
JOÃO
LETICIA
  
```

Fonte: O Autor

Segundo Lidwell (2011), a estética atrai mais as pessoas por fazer o conteúdo parecer mais fácil e atrativo e as cores tem papel fundamental nisso, podendo tornar uma interpretação mais estética e também reforçar o significado a ser passado. Ainda, o todo deve ser alinhado e segmentado para ser mais fácil

de lembrar. Na área de multimídia ou programação web, isso se torna um conceito muito importante.

Para criarmos os cards usamos uma ferramenta online chamada *Canva*⁶, uma ferramenta online e gratuita onde apresenta layouts prontos feitos e postados por designers do Brasil e do mundo inteiro para que possa facilitar na criação de peças gráficas, assim se tornando a forma mais prática de fazer um card de qualquer lugar com vários templates e designs.

Dificuldades: A principal dificuldade encontrada para a criação dos cards foi justamente achar um layout onde tudo ficasse de forma eficiente, limpa, nítida e resumida para o melhor entendimento de quem está iniciando agora na linguagem de programação.

Qual foi a aprendizagem: A aprendizagem durante a criação dos cards foi que nós melhoramos na linguagem, se dedicando ainda mais para que saísse um trabalho bem feito, no qual outras pessoas também pudessem aprender e entender o tanto que nós aprendemos de maneira facilitada.

⁶ <https://www.canva.com>

3.3. CRIAÇÃO DE LISTAS DE EXERCÍCIOS

Tempo Estimado		Horas	Atividade
Início	Final		
25/04/19	17/04/19	2	Preenchimento das listas de exercícios

Objetivo: Elaborar listas de questões sobre cada tema proposto pelo professor.

Porque foi feito: As listas de questões foram feitas para testar a habilidade com a linguagem e iniciar a criação de um banco de questões para exercitar a lógicas dos participantes do projeto.

Descrição: Durante a construção da lista cada integrante do projeto buscava quatro questões de cada assunto, assim preenchendo as listas e ao mesmo tempo que procurava, já entendendo um pouco mais sobre a lógica de cada exercício e propor o algoritmo que resolveria o mesmo.

As listas foram criadas em um arquivo compartilhado, onde todos os integrantes teriam acesso e assim foram sendo preenchidas sem os exercícios fossem repetidos. Cada lista teve, em média, 24 questões, sendo todas de assuntos e dificuldades diversas. A seguir, uma imagem do repositório das listas situado no Google Drive.

Figura 6 - Listas de exercício compartilhadas.

[illegible]

Fonte: O Autor.

Dificuldades: Como havia oito integrantes e não podia repetir questões, em alguns temas foi muito a variedade delas era muito escassa, fazendo ser difícil encontrar mais.

Qual foi a aprendizagem: Aperfeiçoamento da linguagem.

3.4. APRESENTAÇÃO DE FERRAMENTAS PARA OBI

Tempo Estimado		Horas	Atividade
Início	Final		
25/04/19	12/07/19	4	Criação de slides

Objetivo: Fazer uma apresentação de slides contendo sites e ferramentas para se estudar.

Porque foi feito: Apresentar as ferramentas que podem ser usadas em prol do estudo para a próxima turma de estágio.

Descrição: Criamos uma apresentação contendo ferramentas que podem ser utilizadas para o estudo da linguagem de Python, explicando o que cada uma dela faz.

Dentre essas ferramentas estava o Neps Academy⁷, onde você pode encontrar todos os conteúdos e detalhes de Python e pode aprender desde o básico, como o que é um algoritmo e como dar os primeiros passos na programação de computadores, até os conteúdos mais avançados com uma abordagem direta e informal.

A Neps Academy conta ainda com um sistema de exercícios, no qual você pode submeter seu algoritmo que soluciona a questão e a plataforma informará se ele está correto ou não. Abaixo, a figura do segundo slide da apresentação, onde é explicado as funções da Neps.

Figura 7 - Slide 2 da apresentação.

⁷ neps.academy

Neps Academy

— — —

No Neps Academy você irá aprender os principais conteúdos de Ciência da Computação.

Aprendizado desde o básico, o que é um algoritmo e como dar os primeiros passos na programação de computadores, até os conteúdos mais avançados.

Resolução de problemas de programação e terá resposta do sistema em tempo real com os exercícios do Neps.



Fonte: O Autor.

Outra ferramenta que pode ser utilizada é o PyCharm, uma IDE para se programar em Python, focada em deixar a programação mais fácil, tentando deixar o código mais limpo e ajudando o usuário com complementos.

Dificuldades: Nenhuma dificuldade apresentada.

Qual foi a aprendizagem: Aperfeiçoamento da linguagem e do uso de apresentações.

3.5. RESOLUÇÃO COMENTADA DAS QUESTÕES

Tempo Estimado		Horas	Atividade
Início	Final		
		5	Responder as questões em grupo.

Objetivo: Resolver as questões feitas nas listas com ajuda do professor e dos integrantes.

Porque foi feito: Tirar as dúvidas dos alunos e revisar o conteúdo aprendido.

Descrição: A resolução era feita com as listas preenchidas feitas. Era sorteado um aluno para fazer algum exercício que o professor escolhesse da lista até que todos fossem sorteados.

Caso houvesse alguma dúvida, os outros alunos que estavam observando ou o professor ajudavam a saná-la. Após a resolução, a questão entra em debate e os alunos propunham formas para que seu código fosse otimizado ou feito com outra linha de pensamento lógico. Assim, todos os participantes do estágio contribuíam e adquiriam conhecimento por meio da resolução dos exercícios em sala.

Esse método de resolução de questão é chamado de coding dojo e se trata de um local de treinamento de programação. Nesse formato de treinamento, os participantes se integram mais para resolver o código e de forma intuitiva e divertida, consigam aprender em conjunto.

Segundo Bonfim (2014), o coding dojo é um ambiente seguro para testar novas ideias, promover o networking e compartilhamento de ideias entre os membros da equipe. Dessa forma os integrantes podem se conhecer melhor e se adequar ao seu ambiente e além de também conhecer o ambiente de trabalho.

Existem alguns estilos para a condução de uma sessão de Coding Dojo, os mais difundidos são o Randori, Kake e Prepared Kata. Os recursos disponíveis e o nível de conhecimento dos participantes determina o tipo de estilo

que será adotado. Também é possível intercalar os métodos utilizados. (RAMOS, 2015).

Figura 8 - Exemplo de uma resolução de lista.

1 - Faça um programa que calcule a média de 4 números informados (tem que usar o while). Lembrando que a média de 4 números é calculado da seguinte forma: $n1+n2+n3+n4/4$:

```
cont = 0
num = 0
while(cont < 4):
    num1 = int(input("digite a nota do aluno"))
    num = num1 + num
    cont = cont + 1
print("A media é", num/4)
```

2 - Crie um algoritmo que peça dois números(num2 tem que ser maior que num1) e some 1 ao primeiro número até ele ficar igual ao número 2 (Alisson)

```
num1=int(input("digite um numero"))
num2=int(input("digite um numero maior do que o anterior"))
while num1 != num2:
    num1=num1+1
    print(num1,num2)
```

Fonte: O Autor.

Acima, uma figura que contém uma das várias resoluções feitas em grupo nesta atividade.

Dificuldades: Nenhuma dificuldade apresentada.

Qual foi a aprendizagem: Tudo o que foi visto nas vídeo-aulas pode ser aplicado nessa listas e pode tirar as dúvidas mais importantes para a escrita certa da linguagem, sendo essa resolução muito importante para o aprendizado.

3.6. CATEGORIZAÇÃO DAS QUESTÕES DA OBI

Tempo Estimado		Horas	Atividade
Início	Final		
		6	Fazer questões das provas

Objetivo: Aprimoramento das habilidades na linguagem da programação e preparação para a prova da OBI.

Porque foi feito: A categorização das questões foi feita para que tivéssemos contato com nível da prova da OBI e também para aprimorar os conhecimentos já obtidos na linguagem.

Descrição: O orientador Kennedy selecionou algumas questões de provas no site oficial da OBI para que pudéssemos resolver e assim melhorar nossas habilidades com a linguagem. No andamento das resoluções quando alguém apresentava dúvidas o professor deslocava-se bancada ajudar a explicar e resolver a questão. O site permite que nós resolvêssemos a questão em um IDLE e depois só fizéssemos o upload para que a automação corrigisse.

De acordo com o site oficial da OBI, que estava anunciando vigésima primeira Olimpíada Brasileira de Informática de 2019, ela é descrita como “Uma competição organizada nos moldes das outras olimpíadas científicas brasileiras, como Matemática, Física e Astronomia. O objetivo da OBI é despertar nos alunos o interesse por uma ciência importante na formação básica hoje em dia (no caso, ciência da computação), através de uma atividade que envolve desafio, engenhosidade e uma saudável dose de competição.

A estrutura das questões é composta por uma contextualização do ambiente do problema, como no exemplo abaixo, o contexto é dado pela natação e suas medalhas. Seguida do que a entrada e da saída devem conter exatamente, pois seu código é corrigido pelo programa, como seus respectivos exemplos. Abaixo um exemplo de questão da OBI.

Figura 9 - Exemplo de questão da OBI

Medalhas

A natação foi um dos esportes mais emocionantes das Olimpíadas do Rio. Houve até uma prova na qual três atletas chegaram empatados, cada um recebendo uma medalha de prata! Normalmente, porém, os três primeiros colocados terminam a prova em tempos distintos e, portanto, temos a distribuição mais comum de medalhas: o nadador que terminou no menor tempo recebe medalha de ouro; o nadador que terminou com o segundo menor tempo recebe medalha de prata; e o que terminou com o terceiro menor tempo recebe medalha de bronze. Neste problema, dados os três tempos distintos de finalização da prova, dos três nadadores que ganharam medalhas, seu programa deve dizer quem ganhou medalha de ouro, quem ganhou prata e quem ganhou bronze.



Entrada

A primeira linha da entrada contém um inteiro T_1 , indicando o tempo em que o nadador 1 terminou a prova. A segunda linha da entrada contém um inteiro T_2 , indicando o tempo de finalização do nadador 2. Por fim, a terceira linha da entrada contém um inteiro T_3 , indicando o tempo em que o nadador 3 terminou a prova.

Saída

Seu programa deve imprimir três linhas na saída. A primeira linha deve conter o número do nadador que ganhou medalha de ouro; a segunda linha, o número do nadador que ganhou prata; e a terceira linha, o número do nadador que levou bronze.

A resolução proposta neste exercício como correta seria essa:

```

t1 = int(input())
t2 = int(input())
t3 = int(input())

if t1 < t2 and t1 < t3:
    print(1)
elif t2 < t3 and t2 < t1:
    print(2)
elif t3 < t1 and t3 < t2:
    print(3)
else:
    print(1)
    print(2)
    print(3)

```

Dificuldades: Ainda como éramos leigos na linguagem não sabíamos muito como usar tal sintaxe para resolver os exercícios.

Qual foi a aprendizagem: Aperfeiçoamento da linguagem.

4. CONCLUSÃO

O python vem crescendo no mundo inteiro. A prova disso são as estatísticas do site mais conhecido de programação do mundo, o StackOverflow. Segundo os gráficos do site, o Python passou todas as linguagens de programação em apenas 6 anos.

Considerando isso, assimilar o aprendizado de uma língua tão popular se torna importante para a função de técnico de informática, onde a experiência para o curriculum vai ser valorizada.

Segundo o próprio site da olimpíada, o objetivo da OBI é despertar nos alunos o interesse por uma ciência importante na formação básica hoje em dia (no caso, ciência da computação), através de uma atividade que envolve desafio, engenhosidade e uma saudável dose de competição.

A experiência obtida em competir na OBI foi muito importante. O evento auxilia a formação de estudantes do ensino médio, quanto a formação de um técnico de informática. A participação simples, com algumas questões fáceis, médias e difíceis, às quais dá ter noção de como é uma competição de programação.

Além dos alunos aprenderem uma nova linguagem, o período de estágio contribuiu para um melhor desenvolvimento dos integrantes que agora possuem uma área mais ampla na área de informática com mais chance de sucesso no mercado de trabalho. Contribuindo também para o pensamento lógico, que foi necessário resolver questões complexas que exigia um maior tempo. Com isso os artefatos produzidos foram armazenados no drive⁸ do projeto e estão lá para consulta.

⁸ <https://drive.google.com/open?id=1k0NYJ2NggQHb1BwfdvkMGdn1TSd-mNBT>

5. REFERÊNCIAS

BONFIM, Márcio. O que é Coding Dojo. *In*: DevMedia [S.l.]. 2014. Disponível em: <https://www.devmedia.com.br/o-que-e-o-coding-dojo/30517>. Acesso em: 26 out 2019.

BORGES, Luiz Eduardo. **Python para Desenvolvedores**. 2ª Edição. Rio de Janeiro: Edição do Autor, 2010.

SOBRE nós. *In*: Canva [S.l.]. 2019. Disponível em: <https://about.canva.com/pt_br/>. Acesso em: 25 ago 2019.

GOMES, Rodolfo Guimarães. Guia Completo de Python. *In*: DevMedia [S.l.]. Disponível em: <https://www.devmedia.com.br/guia/python/37024>. Acesso em: 25 ago 2019.

INSTITUTO FEDERAL DE EDUCAÇÃO CIÊNCIA E TECNOLOGIA CATARINENSE CAMPUS VIDEIRA. Videira, 2019. **Sobre o IFC**. Disponível em: <http://videira.ifc.edu.br/institucional/sobreoifc/>. Acesso em: 10 ago 2019.

LIDWELL, Willian. **Princípios Universais do Design**. Edição 1. Bookman, 2011.

OLIMPIADA BRASILEIRA DE INFORMÁTICA. **Sobre a OBI**. [S.l.]. Disponível em: <https://olimpiada.ic.unicamp.br/>. Acesso em: 15 de ago 2019.