FACULDADE DA INDÚSTRIA IEL

AVALIAÇÃO 1 - ONLINE

DISCIPLINA: PROGRAMAÇÃO DE COMPUTADORES
PROFESSOR: FÁBIO GARCEZ BETTIO CURSO: ENG. DE SOFTWARE (2P)
ALUNO(A): CLÍSTENES GRIZAFIS BENTO MATRÍCULA: 18597

NOTA

INFORMAÇÕES

- As soluções devem ser apresentadas em linguagem C, estruturada;
- Cada questão da prova encontra-se em uma folha, mantenha assim;
- Abaixo de cada questão existe um espaço que deve ser usado para a solução da questão;
- Se a questão solicitar o código fonte, cole-o nesta área;
- Se a questão solicitar o mapa de memória, teste de mesa ou outro desenho, você pode fazer de três formas:
  - Desenhar em um papel, tirar uma foto e colar na área de resposta (cuidado com a qualidade da imagem, ela deve ser nítida);
  - Desenhar no computador em alguma ferramenta e colar a imagem na área de resposta;
  - Desenhar no próprio Word.
- Esta atividade deve ser entregue no AVA na data estipulada pelo professor com o cabeçalho preenchido com os seus dados e as respostas no próprio arquivo;
- Não serão aceitos arquivos extras, como fotos e códigos;
- Cuidado com cópias, totais ou parciais, da internet ou de colegas;
- Será cobrada a correta identação do código, podendo acarretar na perda de 0,5 pontos por questão mal identada.

QUESTÕES

1. Dado o programa abaixo informe: (1,5)
   a. O que será impresso na tela.
   b. O mapa de memória com o endereço real de cada variável, inclusive os ponteiros.
   c. Teste de mesa da execução de cada linha do programa, comentada.

```
void main()
{
   int i, *p;
   i = 3;
   p = &i;
   printf("%d\n",i);
   printf("%x\n",p);
   printf("%x\n",&i);
   printf("%x\n",&p);
}
```

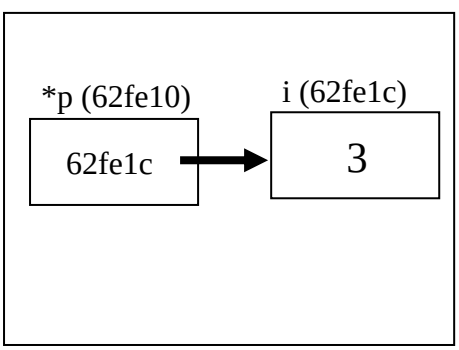
RESPOSTAS:

a) Serão impressos na tela quatro mensagens diferentes, a primeira será o valor da variável inteira i, a segunda é o valor para onde o ponteiro p aponta, a terceira é o endereço da variável i e a última é o endereço do ponteiro p. A imagem seguinte mostra os resultados obtidos através do Dev c++:

```
3
62fe1c
62fe1c
62fe10
```

b) segue abaixo o mapa de memória:

MEMÓRIA RAM



c) Segue abaixo o teste de mesa:

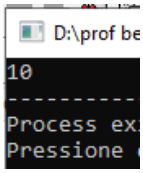
TESTE DE MESA				
STEP	AÇÕES	VARIÁVEIS		
1	Programa iniciado/leitura bibliotecas	i	p	*p
2	Declaração das variável i e o ponteiro *p			
3	Atribuição de valor para variável i	3		
4	Atribuição de valor para o ponteiro p aponta e valor de *p	3	62fe1c	3
5	Impressão do valor de i	3	62fe1c	3
6	Impressão do valor para onde p aponta	3	62fe1c	3
7	Impressão do endereço de i	3	62fe1c	3
8	Impressão do endereço do ponteiro p	3	62fe1c	3
9	Final do programa	3	62fe1c	3

2. Dado o programa abaixo informe: (1,5)
    a. Qual o valor de y no final do programa?
    b. O mapa de memória com o endereço real de cada variável, inclusive os ponteiros.
    c. Teste de mesa da execução de cada linha do programa, comentada.

```
#include <stdio.h>
int main()
{
      int y, *p, x;
      y = 2;
      p = &y;
      x = *p;
      x += 4;
      (*p)++;
      x++;
      (*p) += x;
      printf("%d",y);
      return (0);
}
```

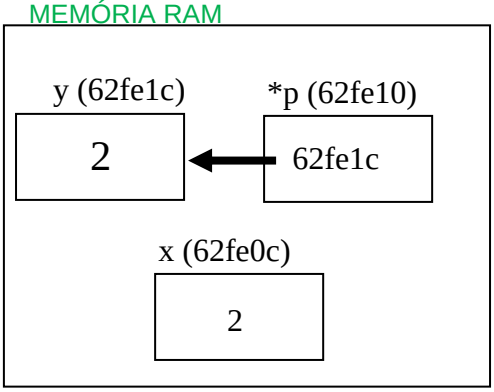
RESPOSTAS

a) O valor de y no final do programa é: 10, conforme resultado no Dev c++:



b) segue abaixo mapa de memória:

MEMÓRIA RAM



c) segue abaixo o teste de mesa:

TESTE DE MESA					
STEP	AÇÕES	VARIÁVEIS			
1	Programa iniciado/leitura bibliotecas	y	p	*p	X
2	Declaração das variáveis y, x e o ponteiro *p				
3	Atribuição de valor para variável y	2			
4	Atribuição de valor para o ponteiro p aponta e *p	2	62fe1c	2	
5	Atribuição de valor para variável x	2	62fe1c	2	2
6	x soma 4	2	62fe1c	2	6
7	*p soma 1	3	62fe1c	3	6
8	x soma 1	3	62fe1c	3	7
9	*p soma x	10	62fe1c	10	7
10	Imprime o valor de y	10	62fe1c	10	7
11	Encerra o programa	10	62fe1c	10	7

3. Crie um pequeno programa em C que: (2,5)
   a. Declare uma estrutura;
   b. Preencha a estrutura com o nome e a idade do usuário;
   c. Imprima o conteúdo da estrutura;

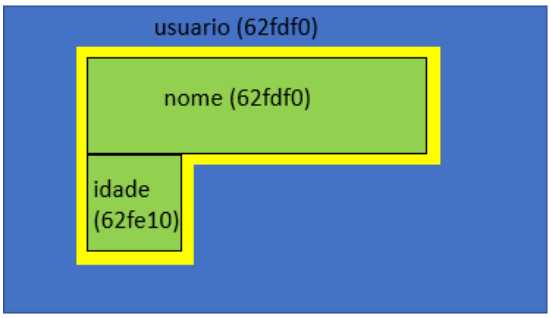
Ao final do programa você deve mostrar:
   a. O código em C da solução;
   b. O mapa de memória com o endereço real de cada variável;
   c. Teste de mesa da execução de cada linha do programa, comentada.

a) seguem abaixo os códigos:

```
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

struct CADASTRO
{
        char nome [30];
        int       idade;
};

int main ()
{
        setlocale(LC_ALL,"portuguese");

        struct CADASTRO usuario;

        printf("Seja bem vindo ao cadasto de usuários");
        printf("\n\nPor gentileza digite seu nome: ");
        scanf("%s", &usuario.nome);
        printf("\nAgora sua idade: ");
        scanf("%d", &usuario.idade);

        system("cls");

        printf("Você se chama %s e ", usuario.nome);
        printf("tem %d anos.", usuario.idade);
```

b) segue abaixo mapa de memória:

MEMÓRIA RAM



c) segue abaixo o teste de mesa:

TESTE DE MESA			
STEP	AÇÕES	STRUCT CADASTRO usuario	
		VARIÁVEIS	
		nome	idade
1	Programa iniciado/leitura bibliotecas		
2	Criação e atribuição do formato da struct		
3	Declaração das variáveis do struct		
4	Início da função int main()		
5	Ajuste de acentuação		
6	Declaração do nome da struct		
7	Mensagem de boas vindas		
8	Mensagem pedindo nome		
9	Leitura do nome	João	
10	Mensagem pedindo idade	João	
11	Leitura da idade	João	30
12	Comando para limpar a tela	João	30
13	Mensagem informando nome e idade	João	30
14	Mensagem informando endereços das variáveis	João	30
15	Fim do programa	João	30

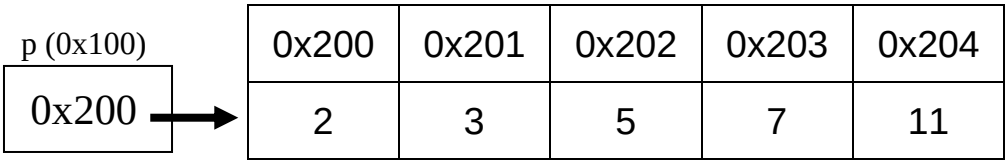
4. Explique a diferença entre p++; (*p)++; *(p++), considere que p é um ponteiro para int com 4Bytes e está no endereço 0x100 da memória (1,5)

Resposta:

- Sendo p um ponteiro que aponta para um inteiro com 4Bytes, p++ é o endereço do inteiro seguinte;
- Já o (*p)++ é o valor do número inteiro *p (para qual o ponteiro aponta) somado a uma unidade;
- *(p++) é o valor do número inteiro para o qual o endereço seguinte de p aponta.

Segue abaixo exemplo para compreensão:

Supondo que o ponteiro p esteja no endereço 0x100 e aponte para um inteiro no endereço 0x200 e a memória está estruturada da seguinte maneira:



Para este caso os valores são:

p = 0x200
p++ = 0x204 (pois a variável inteira nesse caso possuí 4bytes)
(*p)++ = 3
*(p++) = 11
&p = 0x100

5. Crie um programa em C que preencha uma struct COMBO com duas subtructs ITEM usando scanf, depois do preenchimento imprima a struct e todo seu conteúdo. (3,0)

A struct COMBO possui 3 dados.
- sku (inteiro)
- intem1 (struct ITEM)
- intem2 (struct ITEM)

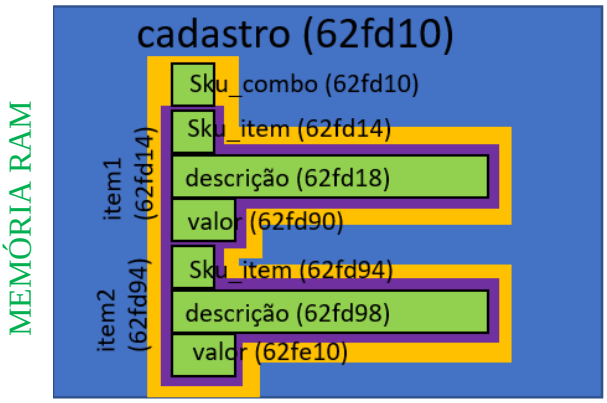
A struct ITEM possui 3 dados.
- sku (inteiro)
- descricao (texto, pode ser String ou vetor de char)
- valor (float)

Ao final do programa você deve mostrar:
a. O código em C da solução;
b. O mapa de memória com o endereço real de cada variável;
c. Teste de mesa da execução de cada linha do programa, comentada.

Resposta:

a) Os códigos estão na próxima página. (goto a;)

b) Segue ao lado o mapa de memória:



c) Segue abaixo o teste de mesa:

TESTE DE MESA								
STEP	AÇÕES	STRUCT COMBO cadastro						
		VARIÁVEIS						
		Sku_combo	Item1			Item2		
			Sku_item	des	valor	Sku_item	des	valor
1	Programa iniciado/leitura bibliotecas							
2	Criação e atribuição do formato struct ITEM							
3	Declaração das variáveis da struct ITEM							
4	Criação e atribuição do formato da struct COMBO							
5	Declaração das variáveis do struct COMBO							
6	Início da função int main()							
7	Ajuste de acentuação							
8	Declaração da struct COMBO cadastro	0	0		0,00	0		0
9	Mensagem de boas vindas	0	0		0,00	0		0
10	Mensagem pedindo sku_combo	0	0		0,00	0		0
11	Leitura da variável sku_combo	1	0		0,00	0		0
12	Mensagem pedindo sku_item, item 1	1	0		0,00	0		0
13	Leitura sku_item1	1	10		0,00	0		0
14	Mensagem pedindo descrição item 1	1	10		0,00	0		0
15	Leitura descrição item 1	1	10	Lápis	0,00	0		0
16	Mensagem pedindo valor item 1	1	10	Lápis	0,00	0		0,00
17	Leitura valor item 1	1	10	Lápis	0,50	0		0,00
18	Mensagem pedindo sku_item, item 2	1	10	Lápis	0,50	0		0,00
19	Leitura sku_item2	1	10	Lápis	0,50	11		0,00
20	Mensagem pedindo descrição item 2	1	10	Lápis	0,50	11		0,00
21	Leitura descrição item 2	1	10	Lápis	0,50	11	Penal	0,00
22	Mensagem pedindo valor item 2	1	10	Lápis	0,50	11	Penal	0,00
23	Leitura valor item 2	1	10	Lápis	0,50	11	Penal	3,00
24	Comando para limpar a tela	1	10	Lápis	0,50	11	Penal	3,00
25	Mensagem informando Sku_combo	1	10	Lápis	0,50	11	Penal	3,00
26	Mensagens informações item 1	1	10	Lápis	0,50	11	Penal	3,00
27	Mensagens informações item 2	1	10	Lápis	0,50	11	Penal	3,00
28	Mensagens informando endereços das variáveis	1	10	Lápis	0,50	11	Penal	3,00
29	Fim do programa	1	10	Lápis	0,50	11	Penal	3,00

a: Códigos do programa

```
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

struct ITEM
{
	int item_sku=0;
	int descricao [30];
	float valor=0;
};

struct COMBO
{
	int combo_sku=0;
	struct ITEM item1;
	struct ITEM item2;
};


int main ()
{
	setlocale(LC_ALL,"portuguese");

	struct COMBO cadastro;

	printf("Seja bem vindo ao cadastro de combo");

	printf("\n\nPor gentileza informe o sku do combo: ");
	scanf("%d",&cadastro.combo_sku);

	printf("\nAgora digite o sku do item 1: ");
	scanf("%d",&cadastro.item1.item_sku);
	printf("\ninforme a descrição do item 1: ");
	scanf("%s",&cadastro.item1.descricao);
	printf("\ninforme o valor do item 1: ");
	scanf("%f",&cadastro.item1.valor);

	printf("\nAgora digite o sku do item 2: ");
	scanf("%d",&cadastro.item2.item_sku);
	printf("\ninforme a descrição do item 2: ");
	scanf("%s",&cadastro.item2.descricao);
	printf("\ninforme o valor do item 2: ");
	scanf("%f",&cadastro.item2.valor);

	system("cls");

	printf("O combo cadastrado com a sku %d possuí os seguintes itens:",cadastro.combo_sku);
	printf("\n\nitem 1: \nsku: %d",cadastro.item1.item_sku);
	printf("\ndescrição: %s", cadastro.item1.descricao);
	printf("\nvalor: %.2f", cadastro.item1.valor);
	printf("\n\nitem 2: \nsku: %d",cadastro.item2.item_sku);
	printf("\ndescrição: %s", cadastro.item2.descricao);
	printf("\nvalor: %.2f", cadastro.item2.valor);


	printf("\n\n\nOs endereços das variáveis são: %x, ",&cadastro); //62fd10
	printf("\n%x, ",&cadastro.combo_sku); // 62fd10
	printf("\n%x, ",&cadastro.item1); //62fd14
	printf("\n%x, %x, %x,  ",&cadastro.item1.item_sku,&cadastro.item1.descricao,&cadastro.item1.valor);
	//62fd14, 62fd18, 62fd90
	printf("\n%x, ",&cadastro.item2); //62fd94
	printf("\n%x, %x, %x,  ",&cadastro.item2.item_sku,&cadastro.item2.descricao,&cadastro.item2.valor);
	//62fd94, 62fd98, 62fe10

	return 0;
}
```