Universidade de Trás-os-Montes e Alto-Douro

Programação Procedimental

Exercícios de Apoio às Aulas Práticas

Engenharia Informática (1º Ciclo)

Tecnologias de Informação e Comunicação (1º Ciclo)

A. Expressões básicas

1. Implemente um programa que permita somar dois quaisquer números, visualizando o resultado obtido.

   Exemplo:

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

int main(void)
{
	int x, y, z;
	/* Requesting the two number from the user */
	printf("Insert two integer values: ");
	scanf("%d %d", &x, &y);
	/* adding the two values */
	z = x + y;
	/* printing the result */
	printf("%d + %d = %d\n", x, y, z);

	system("pause");	//ask the system to pause before exiting
	exit(0);
}
```

2. Implemente um programa que permita calcular a área de um quadrado. (Área=Lado*Lado). O valor do lado deve ser pedido ao utilizador.

3. Implemente um programa que permita calcular o perímetro de um quadrado

4. Implemente um programa que calcule o volume (V) de um cilindro através dos valores da altura (h) do cilindro e do raio (r) da base.
   O volume do cilindro é dado por: V = h * PI * r ^ 2, com PI = π = 3.141592654.

5. Implemente um programa que peça ao utilizador os valores da base e da altura de um triângulo. Em seguida, apresente a área do mesmo.
   Obs.: Área = (Base * Altura) / 2.

6. Implemente um programa que peça ao utilizador um nome completo, e o escreva no ecrã. Considere um máximo de 20 carateres para o nome.
   Exemplo: John Smith

7. O preço de um automóvel é calculado pela soma do preço de fábrica com o preço dos impostos (45% do preço de fábrica) e a percentagem do revendedor (28% do preço de fábrica). Implemente um programa que leia o nome do automóvel e o preço de fábrica e que imprima o nome do automóvel e o preço final.

8. Uma empresa tem para um determinado funcionário uma ficha contendo o nome, número de horas trabalhadas e o número de dependentes de um funcionário. Considere que:
   - A empresa paga 12 euros por hora e 40 euros por cada dependente.
   - Sobre o salário são feitos descontos de 8,5% para o INSS e 5% para IRS.

   Faça um algoritmo para ler o nome, o número de horas trabalhadas e o número de dependentes de um funcionário. Após a leitura, escreva qual o nome, o salário bruto, os valores descontados para cada tipo de imposto e finalmente qual o salário líquido do funcionário.

B. Estruturas de decisão

9. Implemente um programa que peça ao utilizador uma nota de avaliação (numérica) e que indique se o aluno está APROVADO ou REPROVADO.
   Nota: Este exercício pode ser desenvolvido para outras situações como por exemplo ORAL.

   Exemplo:

```c
#include <stdio.h>
int main()
{
	float grade;
	/* insertion of the student's grade */
	printf("Insert the grade: ");
	scanf(" %f", &grade);
	/* presenting the result */
	if (grade >= 9.5) {  // testing the approval conditon
		printf("The student was aproved!\n");
	}
	else {
		printf("The student was not aproved!\n");
	}

	system("pause");
	exit(0);
}
```

10. Elabore um algoritmo que permita calcular a nota Média de um aluno atendendo às notas obtidas nas duas Frequências e visualizar se foi **aprovado** ou **reprovado**, tendo em conta que um aluno aprova sempre que a média é superior ou igual a 9,5 valores.
    O peso das frequências é de 40% para a 1ª e 60% para a 2ª. Para cada uma delas é obrigatória uma nota mínima de 8,5 valores, sem a qual o aluno está automaticamente reprovado.

11. Implemente um programa que converta um par de valores (horas, minutos) do formato 24 horas para o formato AM/PM.

    Exemplo: 13h07 -> 1h07 PM
    00h25 -> 00h25 AM
    12h21 -> 12h21 PM

12. Implemente um programa que converta um valor em bytes para um formato legível (Kilo, Mega, Giga ou Tera bytes consoante o múltiplo que melhor se adapte a uma representação de fácil leitura do valor). 1024 bytes = 1Kbytes

    Exemplo: 16548973 bytes = 15.78 MB

13. Implemente um programa que faça a classificação qualitativa de uma nota (valor inteiro) de um aluno segundo os seguintes níveis:
    - 0 <= nota < 5: péssimo;
    - 5 <= nota < 8: mau;
    - 8 <= nota < 10: insuficiente;
    - 10 <= nota < 12: suficiente;
    - 12 <= nota < 16: bom;
    - 16 <= nota <= 20: excelente;

14. Implemente um programa que indique se um dado número inteiro é PAR ou IMPAR.

15. Implemente um programa que, de entre dois números fornecidos pelo utilizador, permita encontrar o menor deles.

16. Implemente um programa que, de entre três números fornecidos pelo utilizador, permita encontrar o maior deles.

17. Implemente um programa que permita, após a inserção de três valores correspondentes a comprimentos de segmentos de reta, verificar se estes podem formar um triângulo.

18. Implemente um programa que permita, através da inserção da dimensão dos lados de um triângulo, identificar de que tipo de triângulo se trata. Considere apenas as seguintes situações:

    Isósceles - dois lados iguais e um diferente;
    Equilátero - todos os lados iguais;
    Escaleno - todos os lados diferentes.

19. Implemente um programa que permita, através da inserção da dimensão dos lados de um triângulo, verificar se se trata de um triângulo retângulo.
    Sugestão: verificar através do Teorema de Pitágoras efetuando os cálculos com uma aproximação de 0,01.

20. Implemente um programa que permita converter Km em Milhas. Se o número de Km for superior a 5000 deve também visualizar "**muito longe**".
    1 milha = 1609 metros

21. Implemente um programa que faça uma conversão entre Euros e Dólares ou vice-versa consoante a preferência do utilizador. O utilizador deve primeiro escolher o tipo de conversão e depois inserir o valor a converter.
    1,00 € = 1,05 $

C. Estruturas de repetição

22. Implemente um programa que permita aceitar e visualizar de seguida o nome de 20 pessoas. Considere um máximo de 20 carateres para o nome completo.

```c
#include <stdio.h>

#define MAX_NOME 50
#define MAX_NOME_S "49"

void main(void)
{
	char name[MAX_NOME] = "";  // definition and initialization - empty
	int i = 0;

	for (i = 1; i <= 20; i++) {// cycle 20 rounds

		/* insertion of full name */
		printf("Insert the %dº name : ", i);

		//only allows chars between a-z and A-Z + spaces
		//only reads a maximum of MAX_NAME_S chars
		//if there are remaining chars they are kept in the buffer
		scanf(" %" MAX_NOME_S "[a-zA-Z ]s", name);
		// same as: scanf(" %49[a-zA-Z ]s", name);

		/* visualization of the inserted name */
		printf("It was inserted ‘%s’\n", name);
	}
	system("pause");
}
```

23. Implemente um programa que permita aceitar um valor numérico que apenas pode ser positivo. Se não for positivo deve insistir na leitura.

24. Implemente um programa que permita encontrar o maior de 50 números inseridos pelo utilizador.

25. Implemente um programa que leia 30 números inteiros positivos e visualize o menor deles.

26. Implemente um programa que permita encontrar o maior e menor número de uma série de números positivos fornecidos. A sequência termina com o número '0'.

27. Dada uma série de 20 valores reais, implemente um programa que calcule e escreva a média aritmética destes valores. Entretanto se a média obtida for maior que 20 deverá ser atribuído o valor 20 para a média.

28. Implemente um programa que permita efetuar a soma dos 180 primeiros números inteiros positivos.

29. Implemente um programa que permita calcular a soma dos 20 primeiros números pares positivos.

30. Implemente um programa que repita a mensagem "**Introduza a Letra 'a'**" até que se verifique a sua inserção.

31. Implemente um programa que permita calcular o fatorial de um número inteiro, tendo em conta que: **n! = n*(n-1)!; com 0! = 1**
    Exemplo: 4! = 4*3*2*1* 0! = 24;

32. Implemente um programa que verifique se um número é ou não primo.

33. Implemente um programa que, dado um número natural **n**, apresente no ecrã os **n** primeiros números da sequência de *Fibonacci*.

$$f(n) = \begin{cases} 1 & , n = 1 \\ 1 & , n = 2 \\ f(n-1) + f(n-2) & , n > 2 \end{cases}$$

34. Implemente um programa para ler base e altura de 50 triângulos e imprimir a sua área.

35. Implemente um programa que leia as notas de uma turma de 60 alunos numa disciplina e calcule e apresente a média das notas. O professor pode apenas inserir as notas dos alunos que entender, não sendo imperativo que haja a inserção das 60 notas visto que certos alunos podem faltar ao Exame.

36. Implemente um programa que leia os valores anuais de Precipitação de 12 Localidades. Para além disso pretende-se que apresente o valor máximo, o valor mínimo e a média dos valores lidos.

37. Implemente um programa que permita contar todos os números pares e ímpares inseridos pelo utilizador aleatoriamente.

38. Implemente um programa que calcule a média de uma sequência, de valores reais, terminada por -1. O programa deve apresentar igualmente o número de valores introduzidos.

39. Implemente um programa que determine o valor máximo de uma sequência de números inteiros positivos (cuja marca de fim é -1) introduzidos sequencialmente através do teclado. A sequência pode ter 0 (só existe a marca de fim) ou mais elementos.

40. Implemente um programa que mostre dados estatísticos tendo como base um universo de 100 entrevistas realizadas com o objetivo de saber quantas pessoas começaram a trabalhar:
    - com menos de 18 anos;
    - com mais de 18 anos;
    - com menos de 18 anos do sexo masculino;
    - com mais de 18 anos do sexo masculino;
    - com menos de 18 anos do sexo feminino;
    - com mais de 18 anos do sexo feminino.

41. Implemente um programa determine o máximo divisor comum entre dois números inteiros.

42. Implemente um programa determine o mínimo múltiplo comum entre dois números inteiros.

43. Implemente um programa que converta um número de uma base qualquer (entre 1 e 10) para a base 10.
    Ex: 11 na base 2 é igual a 3 na base 10.

D. Funções

44. Implemente um programa que calcule o maior de 3 números reais inseridos pelo utilizador. Implemente para o efeito uma função que calcule/retorne o maior de dois valores reais.

45. Implemente um programa que permita inserir uma sequência de 100 elementos do tipo inteiro com valores entre 0 e 20. Utilize uma função que permita ao utilizador inserir um valor inteiro apenas pode ter valor entre 0 e 20. Se tal não acontecer deve insistir na leitura.

46. Implemente um programa que permita efetuar somas entre dois números fracionários. O resultado deverá ser apresentado no formato de fração. Utilize funções para o máximo divisor comum e para o mínimo múltiplo comum.

47. Implemente um programa de uma máquina de calcular. Esta deverá ter as seguintes funcionalidades:

    a. Um menu para escolher a próxima operação,
    b. As operações básicas: soma, subtração, multiplicação e divisão,
    c. Cálculo do fatorial de um número,
    d. Verificar se um determinado número é primo,
    e. Cálculo da potência de um número,
    f. Calcular o seno ou o cosseno,
    g. Calcular o n.º de dígitos de um número,
    h. Raiz quadrada
    i. Raízes de polinómios de segundo grau

E. **Vetores**

Considere um vetor V com Tamanho N preenchido com números inteiros.

48. Implemente um algoritmo que apresente no ecrã todos os elementos do vetor.

49. Implemente um algoritmo que calcule e apresente no ecrã a soma de todos os elementos do vetor com valor ímpar.

50. Implemente um algoritmo que calcule e apresente no ecrã a soma de todos os elementos do vetor que estejam numa posição de índice ímpar.

51. Crie um programa que preencha um vetor V de N elementos do tipo inteiro. O programa deverá ter pelo menos os seguintes subprogramas:

    a. LerVetor,
    b. MostrarVetor.

52. Crie um programa que armazene num vetor 10 nomes completos.

53. Implemente o algoritmo da ordenação por Seleção.

54. Implemente o algoritmo da ordenação por Bolha.

55. Implemente o algoritmo da pesquisa linear

56. Implemente o algoritmo da pesquisa Binária.

57. Modifique o algoritmo de ordenação por bolha realizado anteriormente de forma a conseguir ordenar um *array* de nomes completos.

58. Escreva o código de uma função que aceite como parâmetro de entrada/saída um vetor com números reais e a dimensão desse vetor. A função deve preencher esse vetor com o valor zero.
    void limpa_vector(float *, int tamanho);.

59. Implemente um algoritmo que armazene uma sequência de números inteiros positivos num *array* dinâmico (cuja marca de fim é -1). Os números devem ser introduzidos sequencialmente através do teclado.

F. **Manipulação de *Strings***

60. Escreva o código de um programa que peça ao utilizador para introduzir o nome e o sobrenome e que construa e apresente no ecrã uma *string* com o nome completo.

61. Escreva o código de um programa que peça ao utilizador para introduzir uma *string* e que apresente no ecrã a *string* escrita "ao contrário".
    Exemplo : "Universidade" deve ser escrito como "edadisrevinU".

62. Escreva o código de um programa que permita a introdução de uma string e que escreva no ecrã a sigla representativa dessa string.
    Exemplo : "World Wide Web" → "WWW".

63. Escreva o código de uma função que aceite como parâmetro de entrada uma *string* e que escreva no ecrã a sigla correspondente a essa *string*. A função deve ignorar as sequências "de", "do", "da", "das", "dos", "e".
    void sigla(char myString[]);
    Exemplo : "Universidade de Trás-os-Montes e Alto Douro" → "UTAD"

64. Escreva o código de uma função que aceite como parâmetro de entrada um texto e uma *string* e que devolva/retorne a quantidade de vezes que a *string* aparece no texto.

G. **Manipulação de ficheiros**

65. Escreva o código de um programa que escreva no ficheiro “mensagem.txt” o texto “Olá mundo!”.

66. Escreva o código de um programa que permita contabilizar o número de linhas de texto que estão armazenadas no ficheiro “dados.txt”.

67. Implemente um programa que calcule todas as potências de base 2 até ao valor 32767, e armazene as mesmas num ficheiro (“pot2.txt”).

68. Implemente um programa converta todos os caracteres minúsculos, de um qualquer ficheiro de texto, em caracteres maiúsculos.

69. Implemente um programa que compare o conteúdo de dois ficheiros e que indique no ecrã se os ficheiros são ou não iguais.

70. Implemente um programa que permita gerir as notas dos alunos referentes a uma avaliação de uma turma. Os alunos são identificados através do número mecanográfico e do nome. A nota deve ter um valor do tipo real.
    A lista de alunos está armazenada num ficheiro “alunos.txt” com a informação número e nome separados pelo carácter ‘#’ (um registo por linha).
    Exemplo: **12456#Maggie Simpson**
    43213#John Doe
    …
    EOF

    O programa deve usar alocação dinâmica das variáveis de grande dimensão e deve incluir a completa gestão da informação tendo um menu de opções com as seguintes tarefas:
    a. Carregar a lista de alunos do ficheiro para a memória;
    b. Guardar a lista de alunos (apenas os alunos) no ficheiro “alunos.txt”, respeitando o formato dos dados especificado para esse ficheiro;
    c. Acrescentar um aluno à lista na memória;
    d. Eliminar, à escolha do utilizador, um aluno da lista na memória;
    e. Atribuir manualmente as notas aos alunos na memória;
    f. Produzir uma pauta em ficheiro com os dados completos de cada aluno. Esse ficheiro, de nome “pauta.txt”, deve conter um registo por linha com os dados separados pelo carácter ‘*’;
       Exemplo: **12480*Laurindinha Ferreira*14.5**
       12576*Jagunço da Silva*7.9

…
EOF

g. Apresentar no ecrã a pauta da avaliação, ordenada por ordem alfabética de nome;
h. Sair do programa;

H. **Listas ligadas**

71. Desenvolva um programa que implemente uma lista ligada (simples) de números inteiros. O programa deve conter funções de manipulação da lista para:
    a. mostrar no ecrã os elementos que estão na lista;
    b. inserir um novo elemento no início da lista;
    c. inserir um novo elemento no fim da lista;
    d. inserir um novo elemento de forma ordenado;
    e. eliminar um elemento da lista;
    f. destruir a lista;.

72. Desenvolva um programa que implemente uma lista de espera de um Centro de Saúde. A implementação da lista de espera deve ser feita recorrendo a uma lista ligada. Para cada elemento da lista deve ser guardado apenas o nome do utente e o número do cartão de utente. Devem ser implementadas as seguintes funções:
    a. mostrar no ecrã os utentes que estão à espera de consulta;
    b. atender o próximo utente (primeiro da lista);
    c. registar novo utente na lista de espera (fim da lista).

73. Implemente um programa, em linguagem de programação C, que permita ao utilizador gerir os salários mensais de uma empresa. Os dados que constituem a folha de pagamentos, respeitantes a cada funcionário, são: o número mecanográfico, o nome, o salário por hora (real) e a quantidade de horas trabalhadas. O programa deve usar listas ligadas (simples) e conter funções para:
    a. Preencher uma folha de pagamento com os dados dos funcionários (criar lista);
    b. Mostrar a folha de pagamentos no ecrã, indicando o salário total de cada funcionário;
    c. Criar um ficheiro “folha.txt” com os dados de todos os funcionários;
    d. Efetuar uma pesquisa, na lista ligada, por um funcionário através do número mecanográfico ou através do nome;
    e. Alterar (na lista ligada) o número de horas trabalhadas referente a um qualquer funcionário (pesquisado com a função da alínea anterior).