Exame de Programação I
(Época normal, duração: 2 horas)

Universidade do Algarve

10 de Janeiro de 2001

- Não é permitido falar com os colegas durante a prova. Se o fizerem, terão a prova anulada.
- Não pode chamar o docente para lhe ajudar a interpretar o enunciado. A interpretação do enunciado faz parte da avaliação.
- A prova tem 6 perguntas e a cotação de cada uma aparece entre parêntesis.

(2 valores) Pergunta 1. Considere o seguinte programa.

```
#include <stdio.h>

main()
{
  int i;
  for( i=0; i<=100; i++ );
    printf("i = %d\n", i);
}
```

Qual destas 4 hipóteses é que é correcta? Justifique a resposta (respostas sem justificação não serão consideradas).

a) O programa tem 1 erro de compilação porque não pode aparecer um ponto-e-vírgula imediatamente a seguir ao `for(...)`

b) O programa dá como output: `i=0`

c) O programa dá como output: `i=100`

d) O programa dá como output: `i=101`

(2 valores) Pergunta 2. Um cientista está a testar um foguete e pretende fazê-lo subir até aos 10000 metros de altura. Na primeira tentativa o foguete subiu até aos 5 metros, na 2.ª tentativa foi até aos 10 metros, na terceira tentativa conseguiu atingir o dobro da altitude da anterior, e assim por diante. Faça uma função em C que ajude o cientista a saber quantas tentativas terá que fazer até o seu foguete chegar aos 10000 metros, ou a qualquer outra altura por ele desejada.

(1,5+3=4,5 valores) Pergunta 3. Observe com atenção o código que se segue:

```
#define MAX_STRING     100
#define MAX_JOGADORES  16

typedef struct {
  char nome[MAX_STRING];
  int idade;
  int internacional;
} atleta;

typedef struct {
  char clube[MAX_STRING];
  char modalidade[MAX_STRING];
  char pais[MAX_STRING];
  atleta jogador[MAX_JOGADORES];
  int num_jogadores;   /* número efectivo de jogadores */
} equipa;

void main (void)
{
   equipa competicao[4];
   ...
}
```

a) Descreva por palavras suas o que `competicao` representa. A descrição deve ser breve, clara, e dada em portugûes corrente de modo a que qualquer pessoa (mesmo que que não saiba nada de programação em C) perceba o seu significado.

b) Imagine que `competicao` tem dados válidos. Escreva um conjunto de instruções em C que permitam listar no ecrã o nome de todos os jogadores de futebol com menos de 20 anos que jogam em equipas portuguesas.

(1,5 valor) Pergunta 4. O problema do ladrão de Faro [1] consiste essencialmente em fazer um programa que calcula todos os subconjuntos de um conjunto de n objectos. Tal pode ser feito, enumerando números binários desde 0 até $2^n - 1$ (ou seja, desde 000...000 até 111...111). Enfim, qualquer coisa deste estilo:

```
...
int i;
int n;            /* numero de objectos */
int nHipoteses;   /* 2 elevado a n */
...
```

[1] para os mais distraídos, este problema foi um dos momentos de avaliação durante o decorrer da disciplina.

```
nHipoteses = pow(2,n);
for( i=0; i< nHipoteses; i++ )
 {
   /* converte 'i' para binário e guarda o resultado numa string */
   ...
 }
```

Este tipo de abordagem funciona bem se n for um número relativamente pequeno. Contudo, se n não for pequeno (por exemplo 100), esta abordagem já não funciona. Porquê?

(1+2+3=6 valores) Pergunta 5. Uma maneira de resolver a limitação anterior, é implementar um contador em binário. Para tal temos de utilizar um array de 0s e 1s. A ideia é inicializar esse array só com zeros. Depois faz-se um ciclo que só termina quando o array estiver só com uns. A cada passo do ciclo incrementa-se o número binário directamente, uma espécie de "binario++" (exemplo com $n = 4$: $0000 \rightarrow 0001$. Outro exemplo: $0011 \rightarrow 0100$). O programa principal ficaria assim:

```
#define MAX_OBJECTOS 1000

main()
 {
   int bin[MAX_OBJECTOS];
   int n;   /* numero de objectos */
   ...
   inic_com_zeros( bin, n );
   while( !tudo_uns( binario, n ) )
    {
      incrementa( bin, n );
      ...
    }
 }
```

a) Implemente a função `inic_com_zeros`.

b) Implemente a função `tudo_uns`.

c) Implemente a função `incrementa`.

(4 valores) Pergunta 6. Faça uma função que elimine elementos repetidos de um array de inteiros. Por exemplo se o array tiver os 5 elementos: $\{1,8,8,3,1\}$, ao tirar os repetidos ficará apenas com 3 elementos: $\{1,8,3\}$. A função deve ter 4 argumentos: (1) array de entrada, (2) número de elementos do array de entrada, (3) array de saída (sem repetidos), e (4) número de elementos do array de saída.