Teste 1 de Arquitetura de Computadores 22/04/2023
Duração 2h (1h30m + 30m)

- *Teste sem consulta e sem esclarecimento de dúvidas*
- *A detecção de fraude conduz à reprovação de todos os envolvidos*

Nº: ***Nome:***

Q1 – Diga quais são as funções do sistema operativo quando um programa está em execução?

Q2 – Qual o conteúdo de um ficheiro executável?

Q3 – Explique porque é que os nomes dos ficheiros estão organizados de forma hierárquica.

Q4 – Diga quais são os recursos atribuídos pelo sistema operativo a um programa quando este está em execução.

Q5 – Diga o que é a redirecção de um canal de saída para um ficheiro e dê um exemplo em que esta funcionalidade do *shell* é útil.

Q6 Na RAM que está atribuída a um programa em execução existem várias zonas. Que zonas são essas e qual o seu papel?

Q7 – Considere o programa em C abaixo.

```
#include <stdio.h>
#include <string.h>
char s[] = “hello, world\n”;
int main() {
       int ls = strlen( s );
       printf(“%d\n”, ls );
       return 0;
}
```

Nas zonas da RAM que identificou na pergunta 6 onde se situam os símbolos seguintes:

main	
ls	
printf	
s	

Q8 – A função C *swap* troca os valores guardados nas variáveis *a* e *b*. Isto quer dizer que se em *a* está guardado o valor 5 e em b guardado o valor 3, depois de invocar *swap* em a ficará guardada 3 e em b ficará armazenado 5. Insira o código que falta na função *swap*.

```
int a = 5;
int b = 3;
void swap( int *x, int *y) {




}
int main{
       swap( &a, &b);
       return 0;
}
```

Q9 Complete o programa seguinte que é invocado com a seguinte linha de comando

```
copia file_old file_new
```

e que copia o conteúdo do ficheiro `file_old` para um ficheiro chamado file_new.

```
#include <stdio.h>
#define MAXBUF 128
int main( int argc, char *argv[] ){
       FILE *f1, *f2;
       char buf[MAXBUF];
       if( argc != 3 ){ printf("Usage: %s file_old file_new\n", argv[0]); exit(1);
       }
       f1 = fopen( ____________________________________);
       if( __________){     perror("file_old"); exit(2);
       }
       f2 = fopen( ____________________________________);
       if( _______){  perror("file_new"); exit(3);
       }
       while (_____________________________________ ) != NULL){
              ____________________________________
       };
       fclose(f1); fclose( f2);
       return 0;
}
```

Q10 Diga o que caracteriza um computador que está organizado de acordo coma arquitetura de Von Neumann

---

Q11 Considere um sistema computacional em que cada posição de memória tem 16 bits. Considerando que na posição de memória M está guardado um número inteiro *sem sinal*, qual o menor e o maior número que pode ser guardado em M? A sua resposta não precisa de indicar um valor, podendo ser uma expressão numérica que inclui uma potência de 2.

Q12– Considere um sistema computacional em que cada posição de memória tem 16 bits. Considerando que na posição de memória M está guardado um número inteiro *com sinal*, qual o menor e o maior número que pode ser guardado em M? A sua resposta não precisa de indicar um valor, podendo ser uma expressão numérica que inclui uma potência de 2.

Q13 – Considere o seguinte programa em C guardado no ficheiro *prog.c.*

```
#include <stdio.h>
int main() {
       char val = 0xFF;

       printf("%d\n", val );
       printf ("%u\n", val);
       return 0;
}
```

Compila-se este programa com o comando `gcc -o prog prog.c` numa arquitetura X86_64. Quando se coloca `prog` em execução o que é que é escrito no *stdout* ? Justifique

---

Q14 Diga em que circunstâncias é que a soma de dois números inteiros com sinal pode dar *overflow*. Explique como é que essa exceção pode ser detetada pelo hardware.

Q15 Explique o ciclo de execução de instruções por um CPU. Qual o papel dos registos PC (Program Counter) e IR (Instruction Register)?

Q16 – Considere o seguinte programa em C guardado no ficheiro *prog.c.*

```
#include <stdio.h>
short int val = 0x0102;
int main() {
       char *p=(char *)&val;
       printf("%d\n", *p );
       printf ("%d\n", *(p+1));
       return 0;
}
```

Compila-se este programa com o comando *gcc -o prog prog.c* e a seguir lança-se *prog* em execução. O que é que é escrito no *stdout* ? Justifique

---

Q17 Na representação de números reais em precisão simples IEEE Floating Point Standard um número real é representado em 32 bits (o bit 31 é o mais significativo e o bit 0 é o menos significativo) com a seguinte interpretação:

- Bit 31: sinal – 0 maior ou igual a zero, 1 menor do que zero
- Bits 30 a 23 (8 bits): expoente E. Sendo E um inteiro sem sinal codificado nestes bits, o valor real do expoente é E - 127
- Bits 22 a 0 (23 bits): mantissa. Sendo a configuração dos bits da mantissa xxxxx...xxx_2, o valor efetivo da mantissa é 1.xxxx...xx_2

Considere o número real -6,25. Preencha a tabela.

Sinal(bit 31)	Expoente (base 10)	Mantissa (base 2)

Use o espaço seguinte para justificar o preenchimento. O preenchimento sem justificação não valerá mais de 20%.

Q18 Considere um CPU coma arquitetura *X86_64*. Suponha que o registo *%rax* contém -5 e o registo *%rbx* contém o valor 3. Qual o conteúdo dos dois registos após a execução da instrução *mov %rax, %rbx*

Q19 Considere um CPU coma arquitetura *X86_64*. Suponha que o registo *%rax* contém -5 e o registo *%rbx* contém o valor 3. Qual o conteúdo dos dois registos após a execução da instrução *subq %rax, %rbx*

Q20 Considere um CPU coma arquitetura *X86_64*. Diga para que serve o registo *%rsp*.