

A stylized illustration of a port scene. In the background, a large cargo ship is docked at a pier. Above the ship, a blue airplane is flying. In the foreground, several trucks are on a road. One truck is carrying a large stack of colorful cargo containers. Another truck is carrying a smaller stack. A third truck is carrying a single container. A forklift is also visible, moving a container. The sky is light blue.

Trabalho de AEDA

Empresa de Transportes e Mercadorias

Componentes:

Allan Borges de Sousa	201800149
Amanda de Olivera Silva	201800698
Juliane de Lima Marubayashi	201800175

DESCRIÇÃO DO PROBLEMA

O problema é dividido em basicamente em três vertentes:

- 1) Adicionar ao programa uma forma de gerenciar de funcionários (motoristas). Assim, ainda deve ser computado o número de horas trabalhadas e a alocação de funcionários para serviços deve ser automática.
- 2) Verificação de inatividade de clientes, ou seja, clientes com mais de um ano sem requisitar serviços são considerados inativos. Deve ser possível, principalmente, pesquisar e visualizar estes.
- 3) Possibilitar o gerenciamento de oficinas de reparo de caminhões. Tais oficinas efetuam serviços e possuem determinada especialidade de marca de automóveis. Ainda, a cada vez que uma oficina aceita um pedido, sua disponibilidade diminui.

DESCRIÇÃO DA SOLUÇÃO

- 1) Os motoristas são gerenciados por uma **BST ordenada por horas de trabalho** composta pela classe **Motorista**. A ordenação foi possível pela implementação do operador \leq , o qual considera a ordem crescente de horas e ordem crescente de nif respectivamente. Para o gerenciamento da BST é feita pela classe Workers.
- 2) A inatividade de clientes é gerenciada por uma **Hash Table com *Clientes**. Como os clientes inativos depende da data do último serviço, foi adicionado à classe **Clientes** um parâmetro do tipo **Data** que armazena a data do último pedido. Assim, sempre que $data\ atual - data\ último\ pedido \geq 1\ ano$ o cliente é considerado inativo. Novos clientes possuem data default = 1500/01/01, no formato yyyy/mm/dd.

Foram utilizados overload do operador () para indicação da igualdade de clientes (nif cliente 1 == nif cliente 2) e para a criação da função hash, que nos caso é o valor do nif.



DESCRIÇÃO DA SOLUÇÃO

3) Uma oficina é representada pela classe **Workshop**, composta por nome, uma lista de marcas nas quais o workshop se especializa e o tempo até que esse workshop possa realizar um novo serviço. Os workshops da empresa são armazenados num vetor na classe **Empresa** devido a necessidade de eventual acesso aleatório e em uma priority queue. A **priority queue** da stl , como é max-heap, está ordenada em ordem decrescente de indisponibilidade. Quando caminhão pede um serviço, é atualizado o workshop no topo da fila. Quando o serviço é específico é criada uma priority queue auxiliar, procura-se um workshop que possua a marca do veículo como especialidade e a priority queue original é atualizada. Para a organização da **priority queue** é usada o overload operator <.

Toda vez que um serviço é requisitado a indisponibilidade de todos os workshops diminui em um, mas a indisponibilidade do workshop requisitado é aumentada. Quanto menor a indisponibilidade, mais “livre” o workshop está.

ESTRUTURA DE FICHEIROS



Motoristas.txt

1	Worker1	Trabalhador
2	2	NIF
3	0	Horas trabalhadas
4	Worker3	
5	3	
6	0	

camioes.txt

1	MARCA1	Marca
2	23	Carga Máxima
3	Congelado	Tipo

clientes.txt

1	Cliente2	Nome
2	201800149	NIF
3	1500/01/01	Data do último serviço
4	Cliente1	OBS:
5	201800175	Data no formato yyyy/mm/dd
6	1500/01/01	

workshops.txt

1	Teste 2	Nome
2	2	Quant. de marcas
3	BMW	Marca1
4	TOYOTA	Marca2
5	20	Indisponibilidade
6	Teste 3	
7	2	
8	HONDA	
9	YAMAHA	
10	30	

servicosx.txt

1	Rio de Janeiro	Saída
2	41	Latitude
3	42	Longitude
4	Petropolis	Destino
5	40	Latitude
6	40	Destino
7	Perigoso	Tipo
8	201800177	Cliente
9	40	Carga máxima
10	Quebravel	Característica
11	3	Camiões

taxas.txt

1		Linha em branco
2		Linha em branco
3	Perigoso	Nome da taxa
4	43	Valor
5	Congelado	Nome da taxa
6	30	Valor

LISTA DE FUNCIONALIDADES - Motorista

As funcionalidades relacionadas a motoristas podem ser acedidas a partir do menu principal opção 10

TRANSPORTATION ENTERPRISE		CLIENTS MANAGEMENT		WORKER MENU	
=====		=====		=====	
Status info. service and client	[1]	Add new client	[6]	Workers MENU and visualization	[10]
Profit info.	[2]	Change client name	[7]	Workshops menu and visualization	[12]
Add truck	[3]	Remove a client	[8]		
Remove truck	[4]	New service request	[9]		
Exit	[5]				
Number of trucks: 5					

Todas as funcionalidades citadas foram completamente implementadas:

WORKERS MENU VISUALIZATION		WORKERS MANAGEMENT	
=====		=====	
First x workers - ascending hours	[1]	Add a worker	[6]
First x workers - descending hours	[2]	Remove a worker	[7]
First x workers - alphabetic order	[3]	Change a worker name	[8]
Search a specific worker by nif	[4]	Reset hours of work	[9]
Cancel	[5]		

LISTA DE FUNCIONALIDADES - clientes inativos

Todas as funcionalidades dos clientes inativos podem ser acedidas a partir do menu principal, opção 1

TRANSPORTATION ENTERPRISE	CLIENTS MANAGEMENT	WORKER MENU
=====	=====	=====
Status info. service and client [1]	Add new client [6]	Workers MENU and visualization [10]
Profit info. [2]	Change client name [7]	Workshops menu and visualization [12]
Add truck [3]	Remove a client [8]	
Remove truck [4]	New service request [9]	
Exit [5]		
Number of trucks: 5		

Todas as funcionalidades citadas foram completamente implementadas:

SEARCH SERVICES	SEARCH CLIENTS
=====	=====
First x most profitable services [1]	First x most profitable clients [6]
First x least profitable services [2]	First x least profitable clients [7]
First x services of a specific type [3]	First x clients in alphabetic order [8]
Specific service by id [4]	Specific client status by nif [9]
Cancel [5]	Show x inactive clients [10]
	Show x inactive clients by date [11]
	Specific inactive status by nif [12]
Option:	

LISTA DE FUNCIONALIDADES - Workshop

As funcionalidade dos workshops podem ser acedidas a partir do menu principal, opção 12:

TRANSPORTATION ENTERPRISE		CLIENTS MANAGEMENT		WORKER MENU	
=====		=====		=====	
Status info. service and client	[1]	Add new client	[6]	Workers MENU and visualization	[10]
Profit info.	[2]	Change client name	[7]	Workshops menu and visualization	[12]
Add truck	[3]	Remove a client	[8]		
Remove truck	[4]	New service request	[9]		
Exit	[5]				
Number of trucks: 5					

Todas as funcionalidades citadas foram completamente implementadas:

WORKSHOPS MENU VISUALIZATION		SERVICE MANAGEMENT	
=====		=====	
Show all workshops	[3]	Add a workshop	[1]
Show earliest available workshop	[6]	Remove a workshop	[2]
		Request generic service for a truck	[4]
		Request specific service for a truck	[5]
Cancel	[7]		

DESTAQUE DE FUNCIONALIDADE

Gostaríamos de destacar a funcionalidade de imprimir motorista por ordem decrescente de horas:

Para a organização da bst em ordem decrescente de horas, a bst.h (arquivo source da estrutura) foi alterado. Houve a implementação de uma nova função chamada **`void getReversedTree(vector<Comparable>& v) const;`** que percorre recursivamente a árvore na ordem reversed pre-order.

```
324 template <class Comparable>
325 void BST<Comparable>::getReversedTree( vector<Comparable> & v) const{
326     getReversedTree_aux(this->root, v);
327 }
328
329 template <class Comparable>
330 void BST<Comparable>::getReversedTree_aux(BinaryNode<Comparable> * t, vector<Comparable> & v) const{
331     if (t != NULL){
332         getReversedTree_aux(t->right, v);
333         v.push_back(t->element);
334         getReversedTree_aux(t->left, v);
335     }
336 }
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
```

```
88 void Workers::printBST_reversed(int n){
89     int counter = 0;
90     vector<Motorista> m;
91     BST_Workers.getReversedTree(m);
92     for (auto const& it: m){
93         counter ++;
94         cout << it;
95         if (counter == n) break;
96     }
97 }
```

PRINCIPAIS DIFICULDADES E ESFORÇO

Neste trabalho, cada componente trabalhou de maneira igual.

As grandes dificuldades do trabalho estão relacionadas em:

- Pensar como os problemas seriam solucionados;
- Na divisão de tarefas dos componentes;
- Aprendizado do sistema de branches do github;
- E principalmente em lidar com erros de escrita em arquivos herdados do projeto anterior e devido a diferença de sistema do grupo. No windows por exemplo, o “enter” é demarcado com “\r\n”, já no linux é usado apenas “\n” e isto desencadeou uma série de erros na formatação dos arquivos e na leitura.